Chapter 2: Input Parameters

by
Dermott E. Cullen
University of California
Lawrence Livermore National Laboratory
L-159

P.O. Box 808
Livermore, CA 94550

Tele: 925-423-7359
E.Mail: cullen1@llnl.gov
Website: http://www.llnl.gov/cullen1

Introduction

This chapter has been updated to define the latest input options and recommendations as of the release of TART2002. Recent updates are indicated in RED, to draw your eye to them.

For a quick reference to all TART input keywords, see the last page of this chapter, which is a complete index of all input keywords. Since in this index all new input options are in RED you can quickly see what options have been added.

WARNING – due to differences in formatting and printers the index to TART input keywords on specific pages of this chapter are only approximate.

Overview of Recent Changes

If you briefly look at the table of TART input keywords at the end of this chapter you will immediately see that the biggest changes in input options are extensions to geometry and how to define it by input. We are always striving to make the definition of geometry both more general and easier for users to prepare input. I'll just briefly mention extensions to third (cubic) and fourth (torus) order surfaces, angular rotation (xrotate), and macro surface (xyzbox, xcan, xconic) and macro volume (xsurf) definitions. Many of the new input options are based directly on user feedback; so don't be shy – if you have a good idea I'd love to hear it.

TART versus TARTNP

TART is a modern, general purpose, neutron-photon code that runs on any computer. The predecessor of TART was the TARTNP code that was designed for higher energy applications, was written in the LRLTRAN language, and only ran on CRAY computers. The TARTNP code is no longer available for use, but every effort

has been made to maintain its capabilities in the modern TART code, so that even older input decks can be used with today's TART code. Where there are differences between TART and TARTNP input I have attempted to briefly document these differences here.

OUTPUT TALLY BINS; MINIMIZE or MAXIMIZE OUTPUT

For neutrons TART is designed to use 700 energy groups between 1.0e-11 MeV and 1 GeV with 50 equally log spaced energy groups per energy decade. However, currently only 616 groups up to 20 MeV are being used. TART will be extended to higher energy when higher energy evaluated neutron data becomes available. For photons TART does not use groups; it is designed to use continuous energy cross sections. Cross sections are defined on a fixed grid of 701 energy points (700 energy intervals) between 1.0e-4 MeV and 1 GeV with 100 equally log spaced energy points per energy decade. Higher energy photon data is currently available, and all 700 energy intervals up to 1 GeV are used.

For both neutrons and photons energy dependent output tallies are limited to a subset of the 616 neutron groups, and 700 photon energy intervals. Standard energy dependent output for neutrons is in 80 energy bins, and for photons in 70 energy bins.

The user has the input option to change tally energy groups, but these cannot be arbitrarily specified. In all cases the boundaries of tally groups MUST be a subset of the 616 neutron group boundaries for neutron output or 700 photon energy intervals for photon output and MUST span the energy range of all particles in a problem. For neutrons you can use sentl 46 to select 80, 175, or 616 energy tally bins. For photons you can use sentl 19 to select 70, 175 or 700 energy tally bins. To further change the tally groups for neutrons use cneutal and for photons use cphotal. These options can be used to specify an arbitrary number of energy groups up to 616 for neutrons or 700 for photons.

If you would like to use a different tally energy bin structure, you have the option to tally results in binary files (e.g., ltype or ltypeg 12) and to then write your own routine to read them and tally them in any manner that meets your needs.

In the other extreme to MINIMIZE output you can use notal to completely eliminate output for SELECTED spatial zones, or sentl 49 to suppress energy dependent output and only output one integral value for ALL zones. These options can be used to minimize the amount of TART output. For example, if you have a problem involved 10,000 zones, but are only interested in energy dependent results in a few spatial zones, use notal to select only the spatial zones that you are interested in. Or if you are not interested in any energy dependent results and only want to know the integral fluence (time integrated flux) and energy deposit in your 10,000 zones use sentl 49.

Overview of Input Preparation

Preparing input parameters for a 3-D combinatorial geometry Monte Carlo code can be very complicated, time consuming and frustrating; this is true not only of this code, but of all codes of this type.

It wasn't too long ago that preparing input for a complicated problem and verifying it could take six months to a year; as I said above, it can be very complicated, time consuming and very, very frustrating. Fortunately today with a little planning and using the diagnostic tools that we now have available input preparation can be greatly simplified and sped up and much of the frustration can be avoided.

There are a number of simple steps that you should follow that can greatly simplify input preparation. These include,

1) spend time to plan exactly what you want to calculate. In particular, you should have a good drawing or blueprint of your geometry.
2) start from a standard, existing input deck and modify it to meet your needs, instead of starting from scratch.
3) use **TARTCHEK**, an interactive graphics program, to check your input, before you try using it with TART.

The most complicated part of input preparation is insuring that the geometry is properly defined. Using **TARTCHEK** can greatly simplify this task for you. There are a number of classic errors that we all occasionally make in input preparation and **TARTCHEK** is designed to allow you to quickly check for these errors.

Somewhat surprisingly, what can also be complicated is properly defining exactly what you want the code to calculate, so that you can properly interpret the output results. For example, if you think the code is calculating influence or path lengths within each zone, when in fact it is actually calculating the leakage crossing zone boundaries, needless to say you will completely misinterpret the results.

TART has an enormous number of options to allow you to score almost anything that you want, by zone, position, direction, energy, time, but in order to obtain the results that you actually want you will have to learn what these options are; these will all be described in detail below.

Step by Step Input Preparation

I recommend that you follow the following steps in preparing input parameters.

Step 1: define your geometry. This involves defining a number of spatial zones, where each zone is defined by the surfaces that bound it. Each zone can only be defined by ANDing together all of the bounding surfaces (some Monte Carlo codes allow zones to be defined by AND and OR operations). For example, you can define a zone as being

outside a cylinder of radius 5 AND inside a cylinder of radius 10 AND above a plane at -5 AND below a plane at +5 (a cylindrical annulus between two planes). But you cannot define a zone as being below a plane at -5 OR above a plane at +5. This convention will sometimes require that you define more spatial zones than you would like, but in most cases is not too much of an inconvenience.

Every TART input deck must define ALL of 3-D space. This isn't as complicated as it sounds at first. The input must define all of the spatial zones that you are really interested in. In addition it must define the rest of ALL space as being outside a non-re-entrant surface that does not contain any material. In this case any particle crossing this non-re-entrant surface can never find its way back into the actual geometry of real interest to you, and the particle can be killed; this is convenient way to terminate histories.

For example, if you have a problem involving ten concentric spheres the input deck must define all space within the ten spheres, but it MUST also define ALL space outside of the sphere of largest radius as a zone containing no material, i.e., a void, in order to allow particle histories to be terminated whenever a particle enters this zone.

Step 2: define all the materials that you will use in your problem. The input to TART requires that you define the density (grams/cc) of each material, as well as the composition of each material. The composition of each material can be defined either in terms of the **atom fraction** of each constituent, e.g., for water 2 atoms of hydrogen and 1 atom of oxygen, or the **weight fraction** of each constituent. The constituents themselves are in all cases defined by the ZA of the constituent, e.g., ZA = 1001 for hydrogen and 8016 for oxygen.

For all of the materials that you will use in your problem, get all this material together and check it carefully. Are you sure that you have the correct ZA for each constituent, and the correct atom or weight fractions, and correct overall density for each material?

The last step is to assign a number to each material, e.g., material 1, 2, 3, etc. After you have defined all your materials and assigned numbers to each, these material numbers can be used to assign a material to each spatial zone. Remember material number 0 is the definition of a void, and is the default assignment if you do not assign any positive material number. Any particles entering a void will have their histories terminated, so be sure to assign a material to every zone that is not a void. Use **TARTCHEK** to insure that there are no interior void zones.

Defining materials for use with TART can often be simplified by using relative density. As we will see below, when a material is assigned to each zone it can be defined in terms of the originally defined material and its overall density multiplied by a relative density for each zone. If the relative density for a zone is not defined by input, by default it is 1. However, if we have a problem involving exactly the same material, but at different densities in different zones, we need only define one original material and use the relative density for individual zones to properly define the contents of the zone. For example, if we have air at different altitudes and assume that the composition is the same at all

altitudes, but its density varies, we can define one standard composition for air and use relative density to define the material at each altitude.

Step 3: define all the physical properties of each zone. Each real, interior zone MUST be assigned a material. In addition for each zone we can define its relative density, as described above, and its temperature if thermal scattering is used.

Whenever a particle enters a zone that does not contain any material its history will be terminated. This applies to the outer most bounding zone, described above, but it also applies to any real interior zone, which mistakenly has not been assigned any material. Every interior zone MUST be assigned a material content. If you want to approximate vacuum in a zone you can make the density of the material extremely small, but you MUST assign something to every real zone. Using this convention puts a burden on the user to insure that every zone does contain a material, but it simplifies TART's job of tracking particles and more importantly makes it much easier to verify the accuracy of your geometry. Once you feel that your geometry is properly defined you can quickly use **TARTCHEK** to verify that you have defined all space and that all real interior space contains a material.

Step 4: Decide what type of problem are you going to run. The choices include: static reactivity, dynamic reactivity, or source problems involving incident neutrons, neutron induced photons, or incident photons. Decide what is it you want to measure or score as a result.

The type of problem you are going to run will affect some of the input parameters that you should select. What you want to measure will definitely affect your input parameters.

If you are running a source problem write down a complete description of your source or sources: what type of particles are involved (neutrons and/or photons), spatial position, energy spectrum, angular dependence, time dependence. Remember that each problem starts from the source and the results can only be as good as your description of your source. Any error in your source description will propagate throughout your problem.

Minimum Required Input

The minimum input required for any TART reactivity or source calculation includes,

1) name - a name to identify the problem
2) box and output ID - to direct output (no longer required by TART 2002 and later)
3) surface definitions - to define surfaces bounding each zone
4) zone definitions - to define spatial regions for scoring
5) material definitions - so that material can be assigned to zones
6) assign material to zone - required for all zones
7) source definition - for all problems (reactivity or source)
8) end - to identify the physical end of input for a problem

Here is an example complete input deck for a fission spectrum of neutrons at the origin in a sphere of water 5 cm in radius,

```
name Red's test
box t32 Red's test
* define a sphere of radius 5 cm
sphere  1  5.0
* zone 1 is inside the sphere
* zone 2 is outside the sphere
jb              1     1
jb              2    -1
* define water at an overall density of 1 gram/cc
matl            1   1.0    2   1001    1    8016
* assign water to zone 1 - inside the sphere
matz            1      1
* define a point source at the origin
source1         1    0.0   0.0   0.0
end
```

Everything else, relative density, temperature, weights, what to score, how many histories to run, the energy and angular dependence of the source, etc. all have default values and need not be defined by input. They need not be defined, but it is very important that you understand what all of these defaults are, if you want to be able to properly interpret your results.

For example, if I run the above problem what do I get as output? By default the code will start with a neutron source starting from a point source at the origin (by default it is an isotropic, time independent, fission spectrum). Is this the source that you expected? By default it will track and score both neutrons and neutron induced photons produced, and it will, again by default, score particles entering a zone. The output results for zone 2 (the outside non-re-entrant zone) will be the neutron and photon leakage from the sphere. Since the source starts in zone 1 and no neutrons can leave and re-enter this zone, no neutrons enter the zone and the output will be 0. Is this what you wanted or expected? Would you have properly interpreted the output? In many cases the answer is no. That's why it is important to understand the default values being used, as well as all the scoring options to guarantee that the code runs exactly the problem you want it to run and produces exactly the results that you want.

Allowed Input Parameters

The following sections describe all of the input parameters allowed to be used with TART. The input parameters used with TART are identical to those used with TARTNP, with the exception of a few input options which were judged to be either archaic and out of date or based on experience have demonstrated that they are difficult and dangerous to use. In addition to the input parameters allowed by TARTNP, TART includes extensions to allow a more general description of input; this is mostly for surface definitions.

Input Lines and Statements

Each input line is free form, blank delimited, with up to 80 characters per line. Input lines can be combined by the use of a continuation indicator (see, Continuation below) to define Input Statements. Each Input Statement starts with a TART keyword in the first column, followed by as many input parameters as required to properly define the input option.

Input Fields

Each input line is read as a string of 80 characters (columns 1 through 80), divided into blank delimited field and then each field is interpreted internally.

A blank delimited input field may define text, fixed or floating point numbers. Text can appear at the beginning of an input line to define a TART keyword, or it can appear elsewhere to define a filename, or to define a range of input, e.g., the word **thru** is used for this purpose. Fixed, or integer, input fields may start with a sign (+ or -), followed only by integers. Floating point numbers may start with a sign (+ or -) followed by integers, decimal point (optional), and exponent (option), e.g., a floating point 3 may to input as 3.0, 3, 3.0e+0, 0.3e+1, etc.; all are equivalent.

All fixed and floating point fields are internally converted from characters to numbers and how they are used is defined by the expected syntax of each input line, not the actual contents of the input line. For example, an input field of a single digit 3 may be interpreted either to define an integer or floating point number, depending only on the expected syntax of the type of input being read.

Keywords

All keywords are input in lower case; upper case (i.e., capital) characters are not allowed. For example, to define a plane perpendicular to the x axis the keyword **xplane** is acceptable, but XPLANE, or Xplane are not acceptable keywords.

Mandatory versus Optional Input Parameters

Each Input Statement MUST start with a keyword left adjusted to start in the first column of the line. The keyword is followed by as many parameters as are required to properly define the input option. Some parameters are mandatory; for example, in defining a sphere the user MUST define a positive radius for the sphere. Other parameters may be optional. In describing all keywords mandatory parameters will be listed after the keyword. Any optional parameters will follow the mandatory keywords and will be enclosed in brackets, e.g., [x0 y0].

Comment Lines

Any line that starts with an asterisk (*) in column 1 or c followed by a blank (c) in columns 1 and 2, is a comment line. TART input decks may contain any number of comment lines. All comment lines are merely listed in the output report and have no effect on the execution of any problem. Users are encouraged to include comment lines in their input decks in order to completely describe the physical significance of any or all input parameters.

Warning: since some TART keywords start with the letter c, c cannot be used to indicate comments, unless it is followed by a blank; only lines starting with an asterisk (*) or c followed by a blank (c) are considered to be comment lines.

Blank Lines

The input deck may contain blank lines anywhere, except before a continuation line. Blank lines MUST be completely blank; not just the first few columns followed by text - they MUST be completely blank.

Continuation

Many input lines may be continued to subsequent lines by including an ampersand (&) as the last character of a line. The first and all continuation lines define an input statement. Continuation lines do not start with a TART keyword; they start and continue with input parameters pertinent to the keyword on the first line of the input statement. For example, the following input is correct,

```
matz   17    1   2   3   4   5   6 &
             10 12 15 17
```

The following input is incorrect, since the second line incorrectly starts with the keyword matz,

```
matz   17    1   2   3   4   5   6 &
matz         10 12 15 17
```

Any input line can now be continued onto any number of continuation lines. With earlier versions of TART some input, particularly complicated sources, could not be continued from one line to another, which made input preparation difficult. You will find that being able to continue any input line, it is much easier to prepare input. Some of the following new options, such as cloning, rotation and spatial translation, were recommended by TART users, and are also designed to simplify preparation of TART input. If you have ideas to even further simplify input preparation, I'd love to hear them.

Later Input Supersedes Earlier Input

Sometimes you can simplify your input deck by realizing that later input supersedes earlier input. For example if you have a problem with many spatial zones, and most zones

contain the same material, and only a few contain a different material, you may be able to simplify your input deck by realizing that you can first assign the same (most prevalent) material to all of the zones, and later assign a different material to a few zones. For example, the following is legal input,

```
matz   1   1 thru 890
matz   2   17  36  412   671
```

The first line assigns material number 1 to zones 1 through 890, and the second like assigns material 2 to zones 17, 36, 412 and 671, thereby superseding the earlier material assignment to these zones.

You can also use this approach to start from a simple problem for preliminary calculations and add progressively more detail to see what effect it has on results. For example, if you have fuel surrounded by air, surrounded by cladding, surrounded by water, in your preliminary calculation you could assign fuel to the fuel, air and cladding zones, i.e. one **matz** input line. For subsequent calculations you could first add the air and perform a calculation, and then add the cladding and perform another calculation. With this step to step procedure you would be able to simply determine the importance of each change to your system and at each step you need not modify the initial **matz** input line, since you know that any later input will supersede it.

TART Extensions

In a number of cases simple extensions have been added to some TARTNP keywords. For example, in the case of cones aligned with a major axis, x, y or x, TARTNP insists that the cone also be centered on the aligned axis, e.g., a cone aligned with the z axis MUST be centered on the z axis. TART allows users to specify aligned cones that are not centered on the axis, e.g., for a cone aligned with the z axis, users are allowed to specify x0 and y0 to define the center of the cone. Any TART extensions will follow the mandatory keywords and will be enclosed in brackets, e.g., {x0 y0}. Off axis cones cannot be specified as cone input to TARTNP, but they can be specified by the more complicated **surf** input.

Compatibility with Old TART Input

TART is designed to be completely compatible with old TART input decks, e.g., if you have an old TART deck from 20 or 30 years ago it should still work with today's TART. I say should, because today's TART performs many more checks of input parameters, so that you may find that a TART input deck that ran to completion many years ago, today may not run because today TART may find errors in your input that were not detected by earlier versions of TART.

Recommended Options

Older versions of TART included default values which were not what we would recommend be used today; for years these default values remained unchanged for compatibility with older TART input decks. Starting with TART2002 this convention was changed to use the default values that we consider to be the BEST choices today. Because of this users should be aware that the answers you obtain from TART today may not be the same as the answers you obtained years ago. But assured that we feel confident that if there are any differences today's answers are better. Here's a summary of the default conventions that have been changed.

Thermal scattering – now defined by default as **sentl 39 1** - without turning on thermal scattering, neutrons will continue to slow down to their minimum allowed and end up in a completely unrealistic energy distribution. Even in problems where you think the neutron spectrum will be very fast with no thermal scattering, you have nothing to lose by turning this option on; if there are no thermal neutrons turning on this option will not result in any penalty in running time, but if there are this is the only way to get realistic results. See, **sentl 39**, below for an complete description of other parameters to set to correctly use thermal scattering.

Resonance Self-Shielding using the Multi-band Method – now defined by default as **sentl 20 1** - without turning on the Multi-band method you will be using unshielded multi-group cross sections. The result will be to generally underestimate the mean free path dimensions of systems, and will in most cases give poor results. Turning on this option will increase running time by about 20-30 %, but the improvement in results is generally worth sacrificing the extra running time. Without turning on this option you won't be saving 20-30 % of the running time; you will most likely be wasting 70-80 % of the running time by investing this much time to get unreliable results.

In addition for photon problems (neutron induced photon production and/or independent photon sources) below about 1 MeV, it is recommended that you turn on fluorescence by using the now defined default **sentl 25 1**. Failure to turn on fluorescence at lower energies will overestimate photon energy deposition.

From the Beginning to the End

The first line of a problem MUST use the **name** keyword, the second MUST use the **box** keyword (not required by TART 2002 or later) and the last line MUST use the **end** keyword.

name problem name

The problem name is up to 40 characters in columns 6 through 45. The first six characters must be unique and should not include special characters such as, +/:, etc. These six characters will be used to define the filename of a restart file. As such, if they are not unique, creation of the restart file may overwrite an existing restart file. Note, presently TART does not create restart files, so that this is not currently a problem.

box ann idinfo

Columns 5 through 7 define an output box number and the remaining columns out to column 34 identify the output. In the past **box** was used by the production version of TARTNP to route output files to a computer output box. This is no longer used by TART and is included only for compatibility with the existing production version of TARTNP. TART leaves all output files on disk at the end of a run and does not attempt to copy them for delivery to a computer output box (not required by TART 2002 or later)

end

Indicates the end of the input for a problem. There are no parameters on this line; it only includes the three characters **end**. An **end** line may be followed by input for another problem; an input deck may contain input for any number of problems. **Warning** - a problem after an **end** line will be read ONLY if the continuation **sentl 24** has been set in the input for the current problem; otherwise the code will terminate at the end of the current problem (sentl 24 is not required by TART2002 or later).

Type of problem - Reactivity or Source

A single keyword **critcalc** defines the type of problem. If this input line is included the problem is a reactivity problem, if this input line is not included the problem is a source problem.

critcalc jsettle nrepeat stat [delt]

A reactivity problem, rather than source problem, will be run. If **delt** is input it will be a dynamic reactivity problem, otherwise a static reactivity problem. **jsettle** is the number of settle cycles to run before starting the calculation; it should be at least 10 to 20 for good results. **nrepeat** is the number of repetitions of the calculation to perform after settling (default =1, only run the calculation once). **stat** is the per-cent standard deviation at which to stop each repetition (default 3.0). **delt** is the time step (shakes) to be used in a dynamic reactivity calculation.

WARNING – stat is in per-cent (%) and is interpreted by TART to be a factor of 10 smaller than the value input. For example input of 3. is interpreted as 0.3 %. Use care in defining stat. For a quick and dirty estimate use an input of 3. (0.3 %), and for a very accurate result use 0.1 (0.01 %). Smaller input than this will merely cause TART to needlessly run for a long time – 0.01 % is much less than the accuracy to which we know any critical assembly parameters.

Recommendations: Before running dynamic reactivity calculations it is recommended that you run a static reactivity calculation and then for your dynamic calculation define **delt** to be about one half the removal lifetime found from the static calculation.

For dynamic criticality problems, TART2002 and later versions will start from the input time step **delt** and then search for the most appropriate time step to be used in the calculation. If you have very spatially inhomogeneous geometry the characteristic time for neutrons can vary enormously as the distribution of neutrons starts from some initial guess and then relaxes into the fundamental mode. Generally this makes it difficult for a single time step to be accurately used throughout a dynamic criticality calculation. In contrast the current TART approach can easily handle this situation. To insure true convergence you should monitor the variation of the time step printed by TART and insure that it has converged to a more or less stable value by the end of the settle cycles; if it hasn't, it is suggested that you increase the number of settle cycles and restart the problem.

If you want to insure that the problem runs to convergence insure that **sentl 2** (the maximum number of batches to run) is a large number. The calculation will end when either convergence has been reached (based on the requested standard deviation **stat**) or the maximum number of batches have been run. For quick and dirty results small batches of 500 to 1000 particles per batch (**sentl 3**) are adequate. For very accurate results use more settle cycles (**jsettle**) and larger batches, e.g., 10,000 to 100,000. The total running time to convergence will depend mostly on the total number of histories run (batches times particles per batch), so that except for the overhead during the settle cycles, using larger batches will not greatly increase total running time, e.g., if all other input parameters are the same, using batches of 10,000 instead of 1,000 will not cause the problem to run 10 times as long, because convergence will be reached for a smaller number of batches.

Restrictions: Only neutrons, not photons, are considered during criticality calculations. **weight** is ignored. Other important input for reactivity calculations include: 1) **sentl 2** and **3**, that define the maximum number of batches to run and particles per batch - the combination defines the maximum number of histories in a calculation and as such will directly effect running time - for reactivity problems use a very large maximum number of batches if you want to insure that you converge before exceeding the maximum number of batches. **Warning** - if the maximum number of batches are run before convergence the problem will end and the results will not be reliable, 2) the starting geometric and energy distribution of the source should be as close as possible to the expected equilibrium distribution. The following input cannot be used for reactivity calculations: **ebias, ediscr, maec, maee, maeeh, s9or10e, s11cone, s11cyl, s11dsk, s11sph, sentl 30, source12, source13, source16, source17, timspec**

Example Input: To run a dynamic reactivity problem, using 20 settle cycles (**jsettle = 20**), only perform the calculation once (**nrepeat =1**), to within a statistical accuracy of 1 percent (**stat = 1**), and a time step of .3 shakes (0.003 microseconds) (**delt = .3**), the following input can be used. By not inputting **delta** or setting it to 0, the same input can be used for a static reactivity problem,

```
critcalc   20   1    1     .3
```

Geometry

Inside or Outside Surfaces

TART uses three dimensional (3-D) combinatorial geometry. The coordinate system is (x, y, z) rectilinear space. Each problem is made up of a number of spatial zones. Each spatial zone is defined by the surfaces that bound the zone. In order to uniquely define zones, in addition to defining which surfaces bound each zone you must also define which side of each surface the zone is located on. For example, if we have a simple problem only involving two concentric spheres, sphere 1 and 2, the problem could involve three spatial zones: 1) inside the inner sphere, 2) between the two spheres, 3) outside the outer sphere. In this case the definition of the three spatial zones should specify that the zones are: 1) **inside** the inner sphere, 2) **outside** the inner sphere and **inside** the outer sphere, 3) **outside** the outer sphere. Note, generally it is not sufficient to merely say a surface bounds a zone; you must also specify whether the zone is **inside** or **outside** the surface. For example, if in the previous example we had said that zone 2) is **inside** both spheres, rather than **outside** the inner sphere and **inside** the outer sphere, we would be describing a different spatial region. You might ask why would one possibly do such a thing. You will find that this is a typical error that one can very easily introduce into problems without realizing it. The concept of **inside** and **outside** surfaces is probably the most difficult thing for users of combinatorial geometry codes to get used to and completely understand; even experienced users will occasionally make errors. Therefore it is worth spending some time discussing it in detail.

TART surfaces are either linear or quadratic. Therefore all surfaces can be described by only considering two simply equations,

linear: $a\,(x0 - x) + b\,(y0 - y) + c\,(z0 - z)$

quadratic: $r^2 - [\,a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2\,]$

More recent versions of TART also allow cubic surfaces,

$$(y-y0)^2 + (z-z0)^2 = a(x-x0)^3 + b(x-x0)^2 + c(x-x0) + d$$

and quartic surfaces, as in the case of a torus,

$$[(x-x0)/a]^2 + [(r-c)/b]^2 = 1; \qquad r^2 = (y-y0)^2+(z-z0)^2$$

For any given spatial point (x, y, z) and surface with given parameters (x0, y0, z0, a, b, c, r) to determine whether the point is **inside** or **outside** the surface it is sufficient to insert these parameters into one of the above two equations and solve it. If the answer is **positive** the point is **inside** the surface, if **negative** the point is **outside** the surface, if **zero** the point is **on the surface**. This isn't as complicated as it sounds. Let's consider in detail all of the surfaces that you will have to deal with.

For simple closed surfaces, such as spheres, ellipsoids or cylinders the concept of **inside** and **outside** is fairly intuitive and easy to visualize. If you visualize a sphere, ellipsoid or cylinder you know what **inside** or **outside** means; you are either **inside** or **outside** the object, it's as simple as that. For example, for a sphere the equation is,

$$r^2 - [\,(x0 - x)^2 + (y0 - y)^2 + (z0 - z)^2\,]$$

For any spatial point (x, y, z) **inside** the sphere the result of solving this equation is **positive**, for any point **outside** the sphere it is **negative** and for any point **on the surface** of the sphere it is **zero**. So that for these surfaces the concept of **inside** and **outside** is simple to understand.

For open surfaces, such as cones and hyperboloids, it may not be as intuitively obvious what **inside** and **outside** mean. In these cases the surface will be with respect to a reference axis. Any spatial point (x, y, z) that is closer to the reference axis than the surface is **inside** and any point that is further from the axis is **outside**. Think of a cone or hyperboloid as a solid object between the axis and its surface, e.g., for a cone visualize a drinking cup. Can you visualize whether a drop of water is **inside** or **outside** of your drinking cup? If you can, you will have no problem with these surfaces. If not, you can always solve the equation for the surface. For example, for a circular cone aligned with the x axis the equation is,

$$-[\,-a\,(x0 - x)^2 + (y0 - y)^2 + (z0 - z)^2\,] \qquad r = 0,\ b = c = 1,\ a > 0$$

$$a\,(x0 - x)^2 - [\,(y0 - y)^2 + (z0 - z)^2\,]$$

Pick any (x, y, z) point and solve this equation to define whether or not a point is **inside** or **outside** the surface.

For the last example, consider a plane perpendicular to the x axis at x = x0, the equation of the surface is,

$$x0 - x = 0$$

All spatial points (x, y, z) with x0 > x will be **inside** the surface (since x0 - x > 0), and all points with x0 < x will be **outside** the surface (since x0 - x < 0), and all points with x0 = x will be **on the surface**.

For planes you may prefer to think in terms of **below** and **above** instead of **inside** and **outside**. I encourage you not to do this. For simple aligned planes it may be easy for you to think of **above** and **below**, but for general rotated planes this can be very confusing. This is also true of general cones. The reason is that in these cases the equations are,

plane: $a\,(x0 - x) + b\,(y0 - y) + c\,(z0 - z)$

cone: $-[a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2\,], \quad r = 0,$
two coefficients, a, b, c > 0, one < 0

Note, that by changing the sign of all of the coefficients, a, b, c, we are defining exactly the same surface, but if we solve these equations the meaning of **inside** and **outside** based on the sign of the answer is completely reversed.

Fortunately only in the case of specifying general planes and quadratic surfaces will you be faced by this inconsistency; in all others cases the program will automatically define the sign of all coefficients. Try to remember to define the coefficients a, b, and c, so that **inside** and **outside** make the most sense to you.

If you can understand the concept of **inside** and **outside** for planes, spheres, cylinders, cones, ellipsoids, hyperboloids, you are in, because those are all of the surfaces that you need understand when dealing with TART. If you don't completely understand the concept don't worry about it; that's the whole purpose of the **TARTCHEK** code, which will completely check your description of geometry and tell you whether or not you have made an error in specifying **inside** or **outside**. If you are not sure of the sign of a surface, rather than worry about it, bang in anything, run **TARTCHEK** and let this interactive graphics program immediately tell you whether or not you have made the right choice.

Preferred axis of orientation

TART uses general (x, y, z), 3-D geometry. However, it has a preferred axis of orientation = the z axis; this is strictly for historical reasons. For general 3-D problems there is no advantage to using any preferred axis of orientation. However, for problems that have one or more axii of symmetry there is a definite advantage when using TART to defining one axis of symmetry to be the z axis. When this is done: 1) you will be able to use all of the geometry keywords, in particular **surfr**, **surfp** and **srotate**, 2) TART can calculate the volume and mass of all zones that are symmetric about the z axis. For general 3-D problems or problems in which you use an axis of symmetry other than the z axis TART cannot calculate volumes and mass of zones.

Therefore it is recommended that if your problem has an axis of symmetry, you define this to be the z axis of your geometry. For example, if you have a number of concentric cylinders whose common central axis are all oriented in the same direction, in general it is completely arbitrary what coordinate system you use and what direction you define this central axis to be. However, for use with TART it is recommended that you define this to be the z axis. Similarly if you have a number of parallel planes, it is recommended that you define them to be perpendicular to the z axis.

Definition of Surfaces

Surface keywords, Number and Parameters

The definition of each surface starts with a TART keyword to define the type of surface, e.g., plane, sphere, cylinder, etc. The keyword is always followed by a surface number; these surface numbers will be used later to define which surfaces bound each spatial zone. The surface number is followed by surface parameters, e.g., the radius of a sphere and position of its center (x0, y0, z0).

First Degree (Planar) Surfaces

All planar surfaces are defined by the first degree equation,

dcx (x0 - x) + dcy (y0 - y) + dcz (z0 - z) = 0

TART allows a general planar surface, plus three special cases of planes aligned with the x, y or z axis. The three special cases correspond to,

1) x plane: dcx = 1, dcy = 0, dcz = 0
2) y plane: dcx = 0, dcy = 1, dcz = 0
3) z plane: dcx = 0, dcy = 0, dcz = 1

xplane nb x0

a planar surface perpendicular to the x axis at x0. The equation of the surface is,

x0 - x = 0

nb - Surface Number
x0 - x coordinate of the plane

yplane nb y0

a planar surface perpendicular to the y axis at y0. The equation of the surface is,

y0 - y = 0

nb - Surface Number
y0 - y coordinate of the plane

zplane nb z0
plane nb z0

a planar surface perpendicular to the z axis at z0. The equation of the surface is,

z0 - z = 0

nb - Surface Number
z0 - z coordinate of the plane

genplane nb x0 y0 z0 [dcx dcy dcz]
gpl nb x0 y0 z0 [dcx dcy dcz]

a general plane passing through the point (x, y, z) = (x0, y0, z0), with the vector normal to the surface (dcx, dcy, dcz). (dcx, dcy, dcz) are optional input parameters; if not specified they default to (0, 0, 1), which corresponds to a zplane. The equation of the surface is,

dcx (x0 - x) + dcy (y0 - y) + dcz (z0 - z) = 0

nb - Surface Number
x0 - x coordinate of the plane
y0 - y coordinate of the plane
z0 - z coordinate of the plane
dcx - direction cosine to x axis
dcy - direction cosine to y axis
dcz - direction cosine to z axis

TART2002 Update

Macro Surfaces and Zones

Starting with TART2002, input can include macro surfaces, which are actually combinations of simpler surfaces; these are convenient to use to define your geometry. So far the list of macro surfaces includes a parallelepiped (**a box**), defined by 6 bounding planes, and a finite length cylinder (**a can**), defined by a cylinder and 2 bounding planes.

In addition to these macro surfaces, input now also includes macro zones, where **xsurf, ysurf,** or **zsurf** input can be used to define many zones and bounding surfaces in a compact form for axially symmetric zones. **addzone** input can be used to add to the list of bounding surfaces for zones defined by **xsurf, ysurf** or **zsurf** input to extend them to non-axially symmetric geometry.

xyzbox nb dx dy dz [x0 y0 z0]

Defines a 3-D parallelepiped by defining its center (x0,y0,z0) and the half width of its three dimensions (dx,dy,dz).

xyzbox2 nb x1 x2 y1 y2 z1 z2

Defines a 3-D parallelepiped by defining its limits in the x, y and z directions.

For xyzbox input is,

nb - Surface Number
dx - half x width of box (center to edge in x direction)
dy - half y width of box (center to edge in y direction)
dz - half z width of box (center to edge in z direction)
x0 - x coordinate of the center of the box
y0 - y coordinate of the center of the box
z0 - z coordinate of the center of the box

For xyzbox2 input is,

nb - Surface Number
x1 - x coordinate of the lower x limit of the box
x2 - x coordinate of the upper x limit of the box
y1 - y coordinate of the lower y limit of the box
y2 - y coordinate of the upper y limit of the box
z1 - y coordinate of the lower z limit of the box
z2 - y coordinate of the upper z limit of the box

Starting from xyzbox input the limits of the box are defined as,

x1 = x0 – |dx| to x2 = x0 + |dx|
y1 = y0 – |dy| to y2 = y0 + |dy|
z1 = z0 – |dz| to z2 = z0 + |dz|

Note, that these are the equivalent xyzbox2 input parameters.

Example #1,

Define surface # 3 to be a 3-D parallelepiped centered on the point (x,y,z) = (6,7,8), with the half widths (dx,dy,dz) = (3,2,3). This means the (x,y,z) limits of the box are,

x0 – dx to x0 + dx = 6 – 2 to 6 + 2 = 4 to 8
y0 – dy to y0 + dy = 7 – 3 to 7 + 3 = 4 to 10
z0 – dz to z0 + dz = 8 – 4 to 8 + 4 = 4 to 12

Input should be,

```
          #   dx   dy   dz    x0   y0   z0
xyzbox    3   2.    3.   4.    6.   7.    8.
```

or,

```
           #   x1   x2   y1   y2   z1   z2
xyzbox2    3   4.   8.   4.   10.  4.   12.
```

Example #2,

If we have 2 nested cubes centered on the origin (0,0,0), one cube 4 cm on a side (2 cm half side) and the second 6 cm on a side (3 cm half side), the following TART input will define 3 zones, 1) inside the inner cube, 2) between the 2 cubes, and 3) outside the outer cube,

```
            #    dx   dy   dz
xyzbox      1    2.   2.   2.
xyzbox      2    3.   3.   3.
```

or,

```
             #    x1   x2   y1   y2   z1   z2
xyzbox2      1    -2.  2 .  -2.   2. –2.   2
xyzbox2      2    -3.  3.   –3.  3.  -3.   3.
```

In either case followed by the definitions of the three zones,

```
jb        1     1
jb        2    -1     2
jb        3    -2
```

xyzbox and xyzbox2

These are logically equivalent and inside TART are treated identically. The two variants are supplied for users who wish to define a box based on the location of its center and length of its three dimensions (xyzbox) as compared to users who wish to define the box based on the exact three dimensional limits of the box (xyzbox2). You can use both together in input, since as soon as either is read it is internally converted to a common definition.

Note, how much simpler using these keywords is compared to describing this geometry with earlier versions of TART; earlier many more surfaces and zones were required, and now you can have volumes more logically defined, such as one zone defining the volume between two boxes.

An xyzbox or xyzbox2 can be used like any other surface to define the limits of a zone. An xyzbox or xyzbox2 can be translated to another center, using addxyz, and they can be used as a boundary of a reflecting zone.

Restrictions: For xyzbox input the coordinates of the center of the box are optional and default to (x0,y0,z0) = (0,0,0). For xyzbox2 ALL 6 limits MUST be input. A xyzbox or xyzbox2 cannot be cloned or rotated. All three dimensions of the box MUST be positive; otherwise it wouldn't make any sense in defining the 3D volume of a zone = (x2-x1)(y2-y1)(z2-z1). On input (dx,dy,dz) for xyzbox and the equivalent for xyzbox2 are defined as

non-negative, so that (x1,x2), (y1,y2), and (z1,z2) are forced into increasing numerical order, regardless of the order they appear as input for xyzbox2.

xcan nb rad dx [x0 y0 z0]
ycan nb rad dy [y0 x0 z0]
zcan nb rad dz [z0 x0 y0]

Defines a 3-D cylinder aligned with an axis (x, y, or z), and two bounding planes, by defining the radius of the cylinder, its half length along the cylinder and its center (x0,y0,z0)..

xcan2 nb rad x1 x2 [y0 z0]
ycan2 nb rad y1 y2 [x0 z0]
zcan2 nb rad z1 z2 [x0 y0]

Defines a 3-D cylinder aligned with an axis (x, y, or z), and two bounding planes, by defining the radius of the cylinder, its limits along the cylinder and its center in the other 2 dimensions.

For xcan, ycan, ycan input is,

nb - Surface Number
dx - half x length of xcan (center to edge in x direction) [xcan only]
dy - half y length of ycan (center to edge in y direction) [ycan only]
dz - half z length of zcan (center to edge in z direction) [zcan only]
x0 - x coordinate of the center axis of the can
y0 - y coordinate of the center axis of the can
z0 - z coordinate of the center axis of the can
WARNING – the order of (x0,y0,z0) input is different for xcan, ycan and zcan.

For xcan2, ycan2, zcan2 input is,

nb - Surface Number
x1 - x coordinate of the lower x limit of the xcan2 [xcan2 only]
x2 - x coordinate of the upper x limit of the xcan2 [xcan2 only]
y1 - y coordinate of the lower y limit of the ycan2 [ycan2 only]
y2 - y coordinate of the upper y limit of the ycan2 [ycan2 only]
z1 - y coordinate of the lower z limit of the zcan2 [zcan2 only]
z2 - y coordinate of the upper z limit of the zcan2 [zcan2 only]
x0 - x coordinate of the center axis of the can [not xcan2]
y0 - y coordinate of the center axis of the can [not ycan2]
z0 - z coordinate of the center axis of the can [not zcan2]

Starting from xcan, ycan or zcan input the limits of the can are defined as,

x1 = x0 – |dx| to x2 = x0 + |dx|

y1 = y0 – |dy| to y2 = y0 + |dy|
z1 = z0 – |dz| to z2 = z0 + |dz|

Note, that these are the equivalent xcan2, ycan2 or zcan2 input parameters.

Example #1,

Define surface # 3 to be a zcan of radius 2.5, centered on the point (x,y,z) = (6,7,8), with the half length (dz) = (4). This means the (z) limits of the can are,

z0 – dz to z0 + dz = 8 – 4 to 8 + 4 = 4 to 12

Input should be,

```
        #   rad   dz   z0  x0  y0
zcan    3   2.5   4.    8.  6.  7.
```

or,

```
        #   rad   z1    z2    x0   y0
zcan2   3   2.5   4.     12.   6.   7.
```

Example #2,

If we have 2 nested cans centered on the origin (0,0,0), one can 4 cm in radius and 12 cm in length (6 cm half length) and the second 6 cm in radius and 16 cm in length (8 cm half length), the following TART input will define 3 zones, 1) inside the inner can, 2) between the 2 cans, and 3) outside the outer can,

```
        #   rad   dz   [z0   x0   y0]
zcan    1     4.    6.
zcan    2     6.    8.
```

or,

```
            #   rad    z1    z2   [x0  y0]
xcan2    1      4.   –6.    6.
xcan2    2      6.   –8.    8.
```

In either case followed by the definitions of the three zones,

```
jb       1     1
jb       2    -1    2
jb       3    -2
```

xcan, ycan, zcan and xcan2, ycan2, zcan2

These are logically equivalent and inside TART are treated identically. The two variants are supplied for users who wish to define a can based on the location of its center and length of the cylinder as compared to users who wish to define the can based on the exact limits of the end of the cylinder. You can use both together in input, since as soon as either is read it is internally converted to a common definition.

Note, how much simpler using these keywords is compared to describing this geometry with earlier versions of TART; earlier many more surfaces and zones were required, and now you can have volumes more logically defined, such as one zone defining the volume between two cans.

An can or can2 can be used like any other surface to define the limits of a zone. A can or can2 can be translated to another center, using **addxyz**, and they can be used as a boundary of a reflecting zone.

Restrictions: For **can** input the coordinates of the center axis of the can are optional and default to (x0,y0,z0) = (0,0,0). For **can2** BOTH limits MUST be input. A **can** or **can2** cannot be **cloned or rotated**. The radius and length of a can MUST be positive; otherwise it wouldn't make any sense in defining the 3D volume of a zone. On input (dx,dy,dz) for can and the equivalent for can2 are defined as non-negative, so that (x1,x2), (y1,y2), and (z1,z2) are forced into increasing numerical order, regardless of the order they appear as input for can2.

xconic nb rad1 x1 rad2 x2 [y0 z0]
yconic nb rad1 y1 rad2 y2 [x0 z0]
zconic nb rad1 z1 rad2 z2 [x0 y0]

Defines a 3-D cone aligned with an axis (x, y, or z), and two bounding planes, by defining the radius of the cone at the position of the two bounding planes. The radii cannot be negative. When the vertex of the cone is between two planes, you MUST divide this into two **conics**; one on either side of the vertex, with the vertex as a common end of both conics; see example #2 input below,

For xconic, yconic, yconic input is,

nb - Surface Number
rad1 - radius of the cone at the lower limit of the conic
rad2 - radius of the cone at the upper limit of the conic
x1 - x coordinate of the lower x limit of the xconic
x2 - x coordinate of the upper x limit of the xconic
y1 - y coordinate of the lower y limit of the yconic
y2 - y coordinate of the upper y limit of the yconic
z1 - y coordinate of the lower z limit of the zconic
z2 - y coordinate of the upper z limit of the zconic

x0 - x coordinate of the center axis of the conic [not xconic]
y0 - y coordinate of the center axis of the conic [not yconic]
z0 - z coordinate of the center axis of the conic [not zconic]
WARNING – the order of (x0,y0,z0) input is different for xconic, yconic and zconic.

Example #1,

Define surface # 3 to be a zconic of radius 2.5 at z = 5., and radius 2.4 at z = 6; the center axis is at (x0,y0) = (0,0), so we can omit these coordinates as input. Input should be,

```
          #   rad1  z1  rad2   z2  (x0 y0)
zconic    3   2.5   5.  2.4    6.
```

Example #2,

We can a cone whose vertex is at z = -2., and we want to define a zconic between z = -4., radius 2, and z = 0., also radius 2. Input should be,

```
          #   rad1  z1    rad2   z2  (x0 y0)
zconic    3   2.    –4..  0.    –2.
zconic    4   0.    –2    2.     0.
```

Note, that this is a example where the vertex is between the two bounding planes, so that we have to use two zconic input lines, the first from z=4 to the vertex at z=-2., and the second from the vertex at z=-2. to z=0.

Example #3,

If we have two nested conics centered on the origin. At z = 5., one conic has a radius of 4. and the other 6. At z = 7. one conic has a radius of 5. and the other 8. The following TART input will define 3 zones, 1) inside the inner conic, 2) between the 2 conics, and 3) outside the outer conic,

```
          #   rad1  z1  rad2   z2  (x0 y0)
zconic    3   4.    5.  5.     7.
zconic    4   6.    5.  8.     7.

jb        1     1
jb        2    -1    2
jb        3    -2
```

Restrictions: For **conic** input the coordinates of the center axis of the conic are optional and default to (x0,y0,z0) = (0,0,0). A **conic** cannot be **cloned or rotated**. The ends of the conic (e.g., z1 and z2) may be input in ascending or descending order, but the length of a conic (e.g, z2 – z1), MUST be non-zero; otherwise it wouldn't make any sense in defining the 3D volume of a zone. The radii cannot be negative, and at least one of the

two radii MUST be positive (otherwise there would be no positive 3D volume). If the two radii entered are positive and equal the **conic** will be converted to a **can** with the same axis of symmetry, e.g., a **zconic** becomes a **zcan**.

xsurf nz ns x1 ri1 ro1 &
x2 ri2 ro2 &
x3 ri3 ro3 &
(continued for more points)

ysurf nz ns y1 ri1 ro1 &
y2 ri2 ro2 &
y3 ri3 ro3 &
(continued for more points)

zsurf nz ns z1 ri1 ro1 &
z2 ri2 ro2 &
z3 ri3 ro3 &
(continued for more points)

nz **starting zone number**
ns **starting surface number**
(x1,ri1,ro1) **triplet of coordinates (any number of triplets can be used)**

This is a compact form in which to define bounding surfaces and zones that are axially symmetric about either x (**xsurf**), y (**ysurf**), or z (**zsurf**) axis. With this input you define a list of coordinates along the x, y or z axis, as well as inner (ri) and outer (ro) radii. From this input TART will define a list of bounding surfaces: planes at the listed coordinates along the axis, and cones or cylinder between the tabulated coordinates. It will then define axially symmetric zones bounded by these surfaces, including correct surface signs for the bounding surfaces.

For example, from the above xsurf input (assuming only 3 tabulated points are input), TART will define x planes at x1, x2 and x3, and conex or cylx for the inner radius between (x1,ri1) and (x2,ri2), (x2,ri2) and (x3,ri3), followed by conex or cylx for the outer radius (x1,ro1) and (x2,ro2), (x2,ro2) and (x3,ro3). [conex will be used if the radii of 2 adjacent points differ, and cylx will be used if the 2 radii are identical]. It will then define 2 axially symmetric zones each bounded by 2 xplanes, and the inner and outer radii defined a conex or cylx.

This is an extremely simple example. Generally you can define very complicated surfaces involving hundreds or even thousands of coordinates, see the below example as well as example problems distributed with TART.

Example input

The following input defines surfaces axially symmetric about the x axis (**xsurf** input). Here we define 9 x 3 = 27 zones extending from z = 0 to 15 cm along the x axis. The inner 9 zones contain water and extend from the axis out to the inner surface of a layer of iron (fe). The middle 9 zones are iron (fe) of varying radius, 2 cm in radial thickness. The outer 9 zones contain water and all extend out to a common radius of 16 cm. As defined here these 27 zones completely fill the volume from the axis out to 16 cm, from z = 0 to 15 cm along the x axis. The uniform outer radius of 16 cm allows this section of geometry to be easily inserted into a more complicated geometry starting at a radius of 16 cm.

Note how in this input I started with zone # 1 and surface #1 for the first xsurf input. Based upon my calculation of the number of zones and surfaces created by the first xsurf input, the input for the second xsurf input starts with zone # 10 and surface # 29. Similarly the input for the third xsurf input starts with zone # 19 and surface #57. In the actual, complete TART input deck the zone and surface numbers subsequently used were restricted to insure no overlap with the 27 zones and 84 surfaces defined by the input shown below.

```
* inside surface = 9 zones + 28 surfaces
*         nz    ns      x       ri      ro
xsurf      1    1       0.0     0.0     1.0   &
                        1.0     0.0     2.0   &
                        2.0     0.0     3.0   &
                        3.0     0.0     4.0   &
                        4.0     0.0     5.0   &
                        5.0     0.0     5.2   &
                        6.0     0.0     5.3   &
                        9.0     0.0     5.3   &
                       12.0     0.0     5.3   &
                       15.0     0.0     5.3
* water
matl    1   1.0   2.0   1001   1.0   8016
matz    1    1 thru   9
* ------------------------------------------------------------------------
-
* surface = 9 zones + 28 surfaces
*         nz    ns      x       ri      ro
xsurf     10    29      0.0     1.0     3.0   &
                        1.0     2.0     4.0   &
                        2.0     3.0     5.0   &
                        3.0     4.0     6.0   &
                        4.0     5.0     7.0   &
                        5.0     5.2     7.2   &
                        6.0     5.3     7.3   &
                        9.0     5.3     7.3   &
                       12.0     5.3     7.3   &
                       15.0     5.3     7.3
* iron
matl    2   7.8    1.0   26000
matz    2   10 thru 18
* ------------------------------------------------------------------------
-
* outside surface = 9 zones + 28 surfaces - out to constant radius
```

```
*         nz    ns     x       ri      ro
xsurf     19    57     0.0     3.0    16.0   &
                       1.0     4.0    16.0   &
                       2.0     5.0    16.0   &
                       3.0     6.0    16.0   &
                       4.0     7.0    16.0   &
                       5.0     7.2    16.0   &
                       6.0     7.3    16.0   &
                       9.0     7.3    16.0   &
                      12.0     7.3    16.0   &
                      15.0     7.3    16.0
* water
matz    1   19 thru 27
```

Restrictions: Starting zone and surface numbers MUST be positive. It is the users responsibility to insure that zones and surfaces defined by these commands do not conflict with zone and surface numbers defined elsewhere in a problem (see below, how to calculate how many zones or surfaces will be defined). Coordinates along the axis (x, y or z) MUST be in strictly ascending order, equality not allowed. Radii cannot be negative, and the outer radius cannot be less than the inner radius.

Addzone input: **xsurf, ysurf** and **zsurf** is a very compact form in which to define axially symmetric geometry, but it is restrictive in that it can only be used for axially symmetric geometry. **Addzone** input is designed to allow you to extend **xsurf, ysurf, zsurf** input to other geometries, by defining additional bounding surfaces **ONLY** for zones defined by **xsurf, ysurf, zsurf** input. If used, **addzone** input **MUST** precede the **xsurf, ysurf, zsurf** input that defines the zone; this is so that the **addzone** defined list of bounding surfaces can be appended to the definition of the zone immediately at the time that **xsurf, ysurf, zsurf** defines the zone.

Calculating zone and surface numbers: Each triplet of coordinates after the first will define a zone. For example if 53 triplets are input, 52 zones will be created. For each zone there are 4 bounding surfaces: 2 planes, and 2 radii. However, after the first zone there will be a common plane used by 2 adjacent zones. As a result the number of surfaces defined is 1 plus 3 times the number of zones defined. For example if 52 zones are defined, 1 + 3*52 = 157 surfaces will be defined. If your **xsurf, ysurf, zsurf** input is too complicated for this type of calculation, it is suggested that you input it to TART and see the output file to determine how many surfaces and zones were actually defined.

Second Degree (Quadratic) Surfaces

All second degree surfaces are defined by the equation,

$$a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2 = r^2$$

TART allows a general second degree surface, plus special cases for spheres and ellipsoids, as well as for circular cylinders and cones that are aligned with the x, y or z axis. TART also allows second degree surfaces to be translated and rotated after their initial definition.

Spheres

The equation of all spheres is,

$$(x0 - x)^2 + (y0 - y)^2 + (z0 - z)^2 = r^2$$

Direct Definition of Spheres

The input for the following keywords differ only in the order that the x0, y0, and z0 coordinates are input. In all cases the input defines a sphere of radius r centered at the point (x, y, z) = (x0, y0, z0).

spherex nb r [x0 z0 y0]

spherey nb r [y0 z0 x0]

spherez nb r [z0 x0 y0]
sphere nb r [z0 x0 y0]

nb - Surface Number
r - Radius of the sphere
x0 - x center of sphere
y0 - y center of sphere
z0 - z center of sphere

Restrictions: the radius (r) must be positive. x0, y0, z0 are optional input and default to 0, 0, 0 (the origin of your coordinate system), if not input.

"Two Point" Spheres

A sphere centered on the x, y or z axis may be defined in terms of two points that it passes through. In order to uniquely define a sphere the two points must be different. In each case the program will solve two simultaneous equations to define the radius and center of the sphere. For example, in the case of a sphere centered on the x axis (y0 = z0 = 0), the equation of the surface is,

$$(x0 - x)^2 + y^2 + z^2 = r^2$$

The sphere passes through the two points (r, x) = (r1, x1) and (r2, x2), where r is the radius from the x axis. The program solves,

$$(x0 - x)^2 = r^2 - (y^2 + z^2)$$
$$(x0 - x1)^2 = r1^2$$
$$(x0 - x2)^2 = r2^2$$

to define r and x0.

These two point spheres can be very useful if you want to guarantee that combinations of surfaces meet exactly at any given set of points.

The following keywords differ only in terms of which of the three axes the sphere is centered on, x, y or z.

spher2px nb r1 x1 r2 x2

a sphere centered on the x axis, (y0 = z0 = 0), and passing through the two points (r, x) = (r1, x1) and (r2, x2), where r is the radius from the x axis. The equation of the surface is,

$$(x0 - x)^2 + y^2 + z^2 = r^2$$

The program will solve to define r and x0.

nb - Surface Number
r1, x1- Point the sphere passes through
r2, x2- Point the sphere passes through

Restriction: x1 and x2 must differ.

spher2py nb r1 y1 r2 y2

a sphere centered on the y axis (x0 = z0 = 0), and passing through the two points (r, y) = (r1, y1) and (r2, y2), where r is the radius from the y axis. The equation of the surface is,

$$x^2 + (y0 - y)^2 + z^2 = r^2$$

The program will solve to define r and y0.

nb - Surface Number
r1, y1- Point the sphere passes through
r2, y2- Point the sphere passes through

Restriction: y1 and y2 must differ.

spher2pz nb r1 z1 r2 z2
spher2p nb r1 z1 r2 z2

a sphere centered on the z axis (x0 = y0 = 0), and passing through the two points (r, z) = (r1, z1) and (r2, z2), where r is the radius from the z axis. The equation of the surface is,

$x^2 + y^2 + (z0 - z)^2 = r^2$

The program will solve to define r and z0.

nb - Surface Number
r1, z1- Point the sphere passes through
r2, z2- Point the sphere passes through

Restriction: z1 and z2 must differ.

Ellipsoids

The equation of all ellipsoids is,

$[(x0 - x)/sx]^2 + [(y0 - y)/sy]^2 + [(z0 - z)/sz]^2 = 1$

an ellipsoid centered at the point (x, y, z) = (x0, y0, z0), with semi-axis sx, sy and sz parallel to the x, y and z axes, respectively.

or, in the standard form described above,

$a (x0 - x)^2 + b (y0 - y)^2 + c (z0 - z)^2 = r^2$

$a = 1/sx^2$
$b = 1/sy^2$
$c = 1/sz^2$
$r = 1$

An Ellipsoid may be defined directly or if symmetric about the z axis, in terms of two points that it passes through.

ellipse nb sx sy sz [z0 x0 y0]

an ellipsoid centered at the point (x, y, z) = (x0, y0, z0), with semi-axis sx, sy and sz parallel to the x, y and z axes, respectively.

nb - Surface Number
sx - x semi-axis
sy - y semi-axis
sz - z semi-axis
z0 - z center of ellipsoid
x0 - x center of ellipsoid
y0 - y center of ellipsoid

Restrictions: all of the semi-axis, sx, sy, sz, must be positive. z0, x0, y0, are optional input and default to 0, 0, 0, if not input.

ellip2p nb z0 r1 z1 r2 z2

an ellipsoid centered at z = z0, symmetric around the z axis, (x0 = y0 = 0), and passing through the two points (r, z) = (r1, z1) and (r2, z2), where r is the radius from the z axis. The equation of the surface is,

$$x^2 + y^2 + c\,(z0 - z)^2 = r^2$$

The program will solve two simultaneous equations to define the radius, r, and c.

nb - Surface Number
r1, z1- Point the ellipsoid passes through
r2, z2- Point the ellipsoid passes through

Restrictions: z1 and z2 must differ and the point further from z0 must have the smaller r coordinate.

Circular Cylinders

TART allows three special cases of circular cylinders, where the axis of the cylinder is aligned with one of the three major axes, x, y or z. In these cases the equation of circular cylinders is,

$$a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2 = r^2$$

where two of the coefficients, a, b and c, are positive and equal, the third is zero and r is positive.

For elliptic and/or non-aligned cylinders see **surfr**, **surfp**, and **srotate**, below.

cylx nb r [z0 y0]

a right circular cylinder of radius r, parallel to the x axis, whose axis passes through the point (x, y, z) = (0.0, y0, z0). The equation of the surface is,

$$(y0 - y)^2 + (z0 - z)^2 = r^2$$

nb - Surface Number
z0 - z center of cylinder
y0 - y center of cylinder

Restrictions: r must be positive. z0, y0, are optional input and default to 0, 0, if not input.

cyly nb r [z0 x0]

a right circular cylinder of radius r, parallel to the y axis, whose axis passes through the point (x, y, z) = (x0, 0.0, z0). The equation of the surface is,

$$(x0 - x)^2 + (z0 - z)^2 = r^2$$

nb - Surface Number
z0 - z center of cylinder
x0 - x center of cylinder

Restrictions: r must be positive. z0, x0, are optional input and default to 0, 0, if not input.

cylz nb r [x0 y0]
cyl nb r [x0 y0]

a right circular cylinder of radius r, parallel to the z axis, whose axis passes through the point (x, y, z) = (x0, y0, 0.0). The equation of the surface is,

$$(x0 - x)^2 + (y0 - y)^2 = r^2$$

nb - Surface Number
x0 - x center of cylinder
y0 - y center of cylinder

Restrictions: r must be positive. x0, y0, are optional input and default to 0, 0, if not input.

Circular Cones

TART allows three special cases of circular cones, where the axis of the cone is aligned with one of the three major axes, x, y or z. In these cases the equation of circular cones is,

$$a (x0 - x)^2 + b (y0 - y)^2 + c (z0 - z)^2 = r^2$$

where two of the coefficients, a, b and c, are positive and equal, the third is negative and r is zero.

For elliptic and/or non-aligned cones see **surfr**, **surfp**, and **srotate**, below.

For macro surface cones, see also **xconic, yconic,** and **zconic**.

Direct Definition of Circular Cones

conex nb ang [x0] {y0 z0}
xcone nb ang [x0] {y0 z0}

a right circular cone, symmetric around the x axis, with its vertex at x = x0, and making an angle ang (degrees) with the x axis. The equation of the surface is,

$$a\,(x0 - x)^2 - (y0 - y)^2 - (z0 - z)^2 = 0$$

nb - Surface Number
ang - angle (degrees) between x axis and surface of cone
x0 - x vertex of cone
y0 - y vertex of cone
z0 - z vertex of cone

Restrictions: the angle ang may not be 0 or 90 degrees. x0 is optional input that defaults to 0 if not input. y0 and z0 are TART extensions.

coney nb ang [y0] {x0 z0}
ycone nb ang [y0] {x0 z0}

a right circular cone, symmetric around the y axis, with its vertex at y = y0, and making an angle ang (degrees) with the y axis. The equation of the surface is,

$$-(x0 - x)^2 + b\,(y0 - y)^2 - (z0 - z)^2 = 0$$

nb - Surface Number
ang - angle (degrees) between y axis and surface of cone
y0 - y vertex of cone
x0 - x vertex of cone
z0 - z vertex of cone

Restrictions: the angle ang may not be 0 or 90 degrees. y0 is optional input that defaults to 0 if not input. x0 and z0 are TART extensions.

conez nb ang [z0] {x0 y0}
zcone nb ang [z0] {x0 y0}
cone nb ang [z0] {x0 y0}

a right circular cone, symmetric around the z axis, with its vertex at z = z0, and making an angle ang (degrees) with the z axis. The equation of the surface is,

$$-(x0 - x)^2 - (y0 - y)^2 + c\,(z0 - z)^2 = 0$$

nb - Surface Number
ang - angle (degrees) between z axis and surface of cone
z0 - z vertex of cone
x0 - x vertex of cone
z0 - z vertex of cone

Restrictions: the angle ang may not be 0 or 90 degrees. z0 is optional input that defaults to 0 if not input. x0 and y0 are TART extensions.

"One Point" Circular Cones

Circular cones aligned with one of the major axis, x, y, or z, may also be defined by defining the vertex of the cone and one point that the cone passes through.

conerx nb x0 r1 x1

a right circular cone, symmetric around the x axis, with its vertex at x = x0, and passing through the point (r, x) = (r1, x1), where r is the radius from the x axis. The equation of the surface is,

$$a\,(x0 - x)^2 - y^2 - z^2 = r^2$$

The program will solve for a.

nb - Surface Number
x0 - x vertex of cone
r1,x1 - a point the cone passes through

Restrictions: x0 and x1 must differ.

conery nb y0 r1 y1

a right circular cone, symmetric around the y axis, with its vertex at y = y0, and passing through the point (r, y) = (r1, y1), where r is the radius from the y axis. The equation of the surface is,

$$-x^2 + b\,(y0 - y)^2 - z^2 = 0$$

The program will solve for b.

nb - Surface Number
y0 - y vertex of cone
r1,y1 - a point the cone passes through

Restrictions: y0 and y1 must differ.

conerz nb z0 r1 z1
coner nb z0 r1 z1

a right circular cone, symmetric around the z axis, with its vertex at z = z0, and passing through the point (r, z) = (r1, z1), where r is the radius from the z axis. The equation of the surface is,

$$- x^2 - y^2 + c\,(z0 - z)^2 = 0$$

The program will solve for c.

nb - Surface Number
z0 - z vertex of cone
r1,z1 - a point the cone passes through

Restrictions: z0 and z1 must differ.

"Two Point" Circular Cones

Circular cones aligned with one of the major axis, x, y, or z, may also be defined by defining two points that the cone passes through.

cone2px nb r1 x1 r2 x2

a right circular cone, symmetric around the x axis, and passing through the two points (r, x) = (r1, x1) and (r2, x2), where r is the radius from the x axis. The equation of the surface is,

$$a\,(x0 - x)^2 - y^2 - z^2 = r^2$$

The program will solve for a and x0.

nb - Surface Number
r1,x1 - a point the cone passes through
r2,x2 - a point the cone passes through

Restrictions: x1 and x2 must differ. If r1 and r2 are equal it will be interpreted as an x cylinder.

cone2py nb r1 y1 r2 y2

a right circular cone, symmetric around the y axis, and passing through the two points (r, y) = (r1, y1) and (r2, y2), where r is the radius from the y axis. The equation of the surface is,

$$- x^2 + b\,(y0 - y)^2 - z^2 = 0$$

The program will solve for b and y0.

nb - Surface Number
r1,y1 - a point the cone passes through
r2,y2 - a point the cone passes through

Restrictions: y1 and y2 must differ. If r1 and r2 are equal it will be interpreted as an y cylinder.

cone2pz nb r1 z1 r2 z2
cone2p nb r1 z1 r2 z2

a right circular cone, symmetric around the z axis, and passing through the two points (r, z) = (r1, z1) and (r2, z2), where r is the radius from the z axis. The equation of the surface is,

$$- x^2 - y^2 + c\,(z0 - z)^2 = 0$$

The program will solve for c and z0.

nb - Surface Number
r1,z1 - a point the cone passes through
r2,z2 - a point the cone passes through

Restrictions: z1 and z2 must differ. If r1 and r2 are equal it will be interpreted as an z cylinder.

General Aligned Quadratic Surfaces

A general aligned quadratic surface is defined by the equation,

$$a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2 = r^2$$

There are seven types of real quadratic surfaces possible with this equation. For example, for a quadratic aligned with the z axis and a and b in standard form, positive, the seven types are as follows,

1) Sphere $a = b = c,\ r > 0$
2) Circular Cylinder $a = b,\ c = 0,\ r > 0$
3) Circular Cone $a = b,\ c < 0,\ r = 0$
4) Ellipsoid $c > 0,\ r > 0$

The above four can be defined using the above keywords. The following three can be defined using **surf** and **surfr** keywords.

5) Elliptic Cylinder $c = 0,\ r > 0$
6) Elliptic Cone $c < 0,\ r = 0$

7) Hyperboloid of one sheet $c < 0,\ r > 0$

For non-aligned quadratics **surfr** can be used to define an aligned quadratic that can subsequently be translated and/or rotated using **surfp** or **srotate**, see below .

surf nb r c [z0 x0 y0 a b]

a quadratic surface centered at (x, y, z) = (x0, y0, z0), defined by the equation,

$$a\,(x0 - x)^2 + b\,(y0 - y)^2 + c\,(z0 - z)^2 = r^2$$

nb - Surface Number
r - constant, e.g., a radius
c - coefficient for z term
z0 - z center
x0 - x center
y0 - y center
a - coefficient for x term
b - coefficient for y term

Restrictions: z0, x0, y0, are optional input that default to 0, 0, 0, if not input. a and b are optional input that default a = b = 1.0, in which case the surface is symmetric about the z axis.

surfr nb r c [a b]

a quadratic surface centered at the origin, and symmetric around the z axis, defined by the equation,

$$a\,x^2 + b\,y^2 + c\,z^2 = r^2$$

Note, this is identical to **surf** with x0 = y0 = z0 = 0.

nb - Surface Number
r - constant, e.g., a radius
c - coefficient for z term
a - coefficient for x term
b - coefficient for y term

Restrictions: a and b are optional input that default a = b = 1.0, in which case the surface is symmetric about the z axis.

Any surface that is defined using **surfr** may be subsequently translated and rotated using **surfp** and **srotate**, to define non-aligned quadratics.

New Surfaces

Cubic

xcubic nb x0 y0 z0 d c b a
ycubic nb x0 y0 z0 d c b a
zcubic nb x0 y0 z0 d c b a

a cubic, rotationally symmetric about an axis - the equations are,

xcubic: $(y-y0)^2 + (z-z0)^2 = R(x)^2$
$= a(x-x0)^3 + b(x-x0)^2 + c(x-x0) + d$
ycubic: $(x-x0)^2 + (z-z0)^2 = R(y)^2$
$= a(y-y0)^3 + b(y-y0)^2 + c(y-y0) + d$
zcubic: $(x-x0)^2 + (y-y0)^2 = R(z)^2$
$= a(z-z0)^3 + b(z-z0)^2 + c(z-z0) + d$

nb - Surface Number
x0, y0, z0 - Center coordinates
d, c, b, a - Coordinates of the cubic

The radius along one axis is represented as a cubic. By defining zones using a cubic and planes perpendicular to the axis of the cubic, you can reproduce almost any surface, using different cubic parameters to apply along different intervals of the axis; exactly as we think in terms of performing cubic spline fits.

You can reproduce almost any surface depending on a, b, c and d - spheres, ellipses, cylinders, cones, parabola, hyperbola, plus more complicated shapes.

WARNING it is R^2, NOT R, that is represented by a cubic.

Example problem: NEWCUBIC.IN

Torus

xtorus nb x0 y0 z0 a b c
ytorus nb x0 y0 z0 a b c
ztorus nb x0 y0 z0 a b c

a torus aligned with an axis - the equations are,

xtorus: $[(x-x0)/a]^2 + [(r-c)/b]^2 = 1$
$r^2 = (y-y0)^2+(z-z0)^2$
ytorus: $[(y-y0)/a]^2 + [(r-c)/b]^2 = 1$
$r^2 = (x-x0)^2+(z-z0)^2$

ztorus: $[(z-z0)/a]^2 + [(r-c)/b]^2 = 1$
$r^2 = (x-x0)^2+(y-y0)^2$

nb - Surface Number
x0, y0, z0 - Center coordinates
a, b, c - Coordinates of the torus

If a =b, it is a circular torus, otherwise it is an elliptical torus.

Example problem: NEWTORUS.IN

Restrictions: a and b MUST be positive. If c is positive the shape is a torus, if c = 0 the shape is a sphere, and if x is negative the shape looks like an American football.

Cloning (Duplicating) Surfaces

clones ns is1 thru is2
clones ns is1 is2 is3.....

Clone (copy) a surface any number of times. Surface ns is copied to define surfaces is1 thru is2, or is1 is2 is3.....

ns - surface number to clone (MUST be defined)
is1 thru is2 - make surface numbers is1 thru is2 identical to surface ns
is1 is2 is3.. - make the list of surface numbers identical to surface ns

Limitations: surface number ns MUST be defined BEFORE it can be cloned (copied). The surface numbers is1 thru is2 or is1 is2 is3... MUST NOT be defined.

Any surface may be cloned, any number of times.

This option can be used to minimize input preparation when you have a number of identical surfaces that will finally be located at different locations. You can input a surface once, clone it, and then later translate and/or rotate the clones to their final locations.

WARNING - these options are executed immediately when they are read from a TART input deck. Therefore the surface to be cloned (ns) MUST be defined before it can be cloned.

Example Problems: NEWHEX.IN, NEWROT.IN and NEWROT2.IN

Reduced, Reflecting Geometry

xabove x0
yabove y0

zabove z0

xbelow x0
ybelow y0
zbelow z0

For users who only want to model 1/2, 1/4 or 1/8 of symmetric geometry, these options can be used to: 1) define additional x, y and/or z planes, 2) add these planes as boundaries of ALL zones, 3) add additional, reflecting zones on the "other" side of the planes.

x0, y0, z0 - a plane perpendicular to the axis is defined at one of these coordinates.

"above" means transport above this plane - the reflecting zone is below this plane.

"below" means transport below this plane - the reflecting zone is above this plane.

With earlier versions of TART in order to accomplish this you had to include the surface of the reflecting zone explicitly as a bounding surface of every zone. With this new input option this is automatically done for you.

For TARTCHEK users, to see the effect of inserting these planes, use the above/below options on the "Surface" page.

WARNING: These planes are inserted into the geometry AFTER ALL input has been read - they CANNOT be rotated or transformed in ANY way. It is suggested that as a reminder to yourself, you always locate these options at the end of your TART input deck after all other geometric input parameters have been defined.

Try: Adding this to any of the example input

Translation and Rotation

Surfaces previously defined by **surfr** input may be subsequently translated and rotated. Note, for a collection of objects that all have the same center, as defined by **surfr** input, the following keywords may be used to translate and rotate the entire collection of objects, e.g., define a simply aligned object at the origin and subsequently translate and rotate the entire object. The following keywords are used to translate and rotate surfaces,

surfp nb x0 y0 z0 [dcx dcy dcz]

nb must be a surface previously described by **surfr** input. The center of symmetry, initially at the origin, is translated to the point (x, y, z) = (x0, y0, z0). The axis of the surface, initially the +z axis, is rotated to the direction (dcx, dcy, dcz).

nb - Surface Number
x0 - x center

y0 - y center
z0 - z center
dcx - direction cosine to x axis
dcy - direction cosine to y axis
dcz - direction cosine to z axis

Restrictions: surface number nb must have been previously defined by **surfr** input. (dcx, dcy, dcz) are optional input that default to (dcx, dcy, dcz) = (0, 0, 1), which corresponds to no rotation; this option can be used to translate surfaces, without rotation.

srotate nb nb1 nb2 nb3.....

nb must be a surface previously described by **surfr** input, and assigned a translation and rotation by a **surfp** input. **nb1, nb2, nb3**.....must also have been previously described by **surfr** input. The surfaces **nb1, nb2, nb3**...will be assigned the same translation and rotation as that assigned to **nb**.

This keyword is a convenient means of assigning the same translation and rotation to a collection of surfaces. The same thing can be accomplished by inputting **surfp** lines all with the same translation and rotation parameters for each of the surfaces **nb, nb1, nb2, nb3**.....

nb - A previously defined **surfr** surface number
nb1 - A previously defined **surfr** surface number
nb2 - "
etc.

Restrictions: surface number nb must have been previously defined by both **surfr** and **surfp** input lines. Surface numbers **nb1, nb2**,... must have been previously defined by **surfr** input.

New Rotation and Translation Options

Rotation about the X, Y or Z axis

xrotate ang is1 thru is2
xrotate ang is1 is2 is3.....
yrotate ang is1 thru is2
yrotate ang is1 is2 is3.....
zrotate ang is1 thru is2
zrotate ang is1 is2 is3.....

A rotation of surface(s) about an axis by a clockwise angle **ang** (degrees) looking up the axis. Rotation is about the ORIGIN - not the center of the surface. Note, this differs from **surfp** and **srotate** input, which can only be used to rotate **surfr** input about the center of the surface.

ang - angle of rotation in degrees
is1 thru is2 - rotate surface numbers is1 thru is2
is1 is2 is3... - rotate the listed surface numbers

Surfaces can be rotated one or more times, and the rotation is cumulative and order dependent.

Any linear or quadratic surface may be rotated. Cubic and torus MAY NOT be rotated (at least yet).

WARNING - these options are executed immediately when they are read from a TART input deck. Therefore all surfaces to be rotated MUST be defined before they can be rotated, and the order of rotations is important.

WARNING - for TARTCHEK users, the lower, left hand plot, is looking at the front of your geometry in the (z,x) plane, looking UP THE Y AXIS. The upper, left hand plot, is looking down at the top of your geometry in the (z,y) plane, looking DOWN THE X AXIS. The lower, right hand plot, is looking at the right hand size of your geometry in the (y,x) plane, looking DOWN THE Z AXIS. As a result, a clockwise rotation about the y axis will appear clockwise in the lower, left hand plot. However, a clockwise rotation about the x axis will appear COUNTERCLOCKWISE in the upper, left hand plot, and a clockwise rotation about the z axis will appear COUNTERCLOCKWISE in the lower, right hand plot. This isn't an error - it is merely a result of your perspective when viewing TARTCHEK displays.

Example problems: NEWHEX.IN, NEWROT.IN (1 rotation) and NEWROT2.IN (2 rotations)

Translation of Spatial Coordinates

addxyz xadd yadd zadd is1 thru is2
addxyz xadd yadd zadd is1 is2 is3.....

Add (x,y,z) to the current center of surfaces.

xadd, yadd, zadd - add to the current (x0, y0, z0) center coordinates of surfaces
is1 thru is2 - add to surface numbers is1 thru is2
is1 is2 is3... - add to the listed surface numbers

Any surface may be translated, any number of times - and the results are cumulative.

This can be used to translate an entire object or objects to a new location, by translating all bounding surfaces by the same amount. It also simplifies input by allowing you to ignore the final position of a collection of surfaces, and input them as if they are at the origin - then later "add" their final center coordinates.

WARNING - these options are executed immediately when they are read from a TART input deck. Therefore all surfaces to be spatially translated MUST be defined before they can be spatially translated.

Try: Adding this to any of the example input

Definition of Zones

Each spatial zone is defined by the surfaces that bound the zone. In order to uniquely define zones, in addition to defining which surfaces bound each zone you must also define which side of each surface the zone is located on.

TART uses three different types of zones: 1) normal real interior zones that you are interested in; these MUST contain material, 2) an exterior non-re-entrant zone (the rest of the universe); this MUST not contain material, and 3) reflecting zones; these MUST not contain material. The first two types of zones are defined using **jb**, **zonejb and zone** or **bjp** keywords; these are described immediately below. Reflecting zones will be defined after describing **jb**, **zonejb, zone** and **bjp**.

Interior and Exterior Zones

Each problem may use one of two possible ways to define zones: you can either use **jb** or **bjp** input, but you cannot mix the two together in a single problem. With **jb** input you define each of the bounding surfaces and a sign to say whether the zone is inside or outside of the surface. With **bjp** input for each bounding surface you define the same information as with **jb** input, plus you define the most probable zone that a particle will enter if it leaves the current zone across this surface. With **jb** input the code will learn as it progresses through a problem what zone a particle will enter if it leaves the current zone across this surface and optimize the zone search strategy accordingly. With **bjp** input the code will always assume that the new zone entered will be exactly the one indicated by input and it will not attempt any further optimization. The **recommended** option is to use **jb** input; **bjp** input should only be used in special cases, as will be explained in detail below.

jb	**nz**	**nb1**	**nb2**	**nb3.....[&]**
zonejb	**nz**	**nb1**	**nb2**	**nb3.....[&]**
zone	**nz**	**nb1**	**nb2**	**nb3.....[&]**
addzone	**nz**	**nb1**	**nb2**	**nb3.....[&]**

nz is a zone number (1 to MAXZONE) and **nb1, nb2, nb3,....**are signed surface numbers. Blank or plus as the first character of **nb1, nb2,** etc., indicates that the zone is **inside** the surface, **minus** indicates the zone is **outside** the surface.

For example, if I have two spherical surfaces of different radii and three zones: 1) **inside** the inner sphere, 2) **outside** the inner sphere and **inside** the outer sphere, 3) **outside** the outer sphere, the definition of the three zones could be,

```
* Definition of 2 spherical surfaces
sphere 1        5.0
sphere 2        8.0
* Definition of 3 zones by their bounding surfaces
jb      1       1
jb      2      -1       2
jb      3      -2
```

This first defines surfaces 1 and 2 as spheres of radius 5 and 8 cm, respectively, centered on the origin (default x0, y0, z0 = 0, 0, 0). It next defines zone 1 to be inside the sphere of radius 5, zone 2 to be outside the sphere of radius 5 and inside the sphere of radius 8, and zone 3 to be outside the sphere of radius 8.

The input line may be continued to any number of successive lines by ending a line with an ampersand (&). Continuation lines do not include the keyword, jb, zonejb or zone, or the zone number nb; they only contain signed surface numbers. For example, if zone 1 is bounded by surfaces 6 through 18 (with various signs) the input could look like the following,

```
jb    1     6    -7   8    -9   10   -11   12  &
           13  -14  15   -16   17   -18
```

If the zone were bounded by more surfaces additional continuation lines can be used (as many as are required to define all bounding surfaces).

Restrictions: this option cannot be used when **bjp** input is used, see below. All of the surface numbers used as boundaries of each zone must all be defined by input. **addzone** input can only be used to add to the list of bounding surfaces for zones defined by **xsurf, ysurf** or **zsurf** input. If used, **addzone** MUST precede the corresponding **xsurf, ysurf** or **zsurf** input. See the example problems distributed with TART.

bjp nz nb1 mp1 nb2 mp2 nb3 mp3.....[&]

This is an infrequently used option to describe zones, and is only used for special applications, usually involving overlapped geometry, as explained in detail below.

Restrictions: this option cannot be used when **jp** input is used, see below. All of the surface numbers used as boundaries of each zone, and all most probable zones must be defined by input.

nz is a zone number (1 to MAXZONE), **nb1, nb2, nb3,....**are positive surface numbers and **mp1, mp2, mp3,.....**are signed most probable zone numbers that the particle will

enter when it crosses surfaces **nb1, nb2, nb3...**, respectively. Blank or plus indicated that the zone is **inside** the surface, **minus** indicates the zone is **outside** the surface.

For example, if I have two spherical surfaces of different radii and three zones, two of which overlap: 1) **inside** the inner sphere, 2) **inside** the outer sphere, 3) **outside** the outer sphere, the definition of the three zones could be,

```
* Definition of 2 spherical surfaces
sphere 1        5.0
sphere 2        8.0
* Definition of 3 zones by their bounding surfaces
bjp     1       1    2
bjp     2       2    3
bjp     3       2   -2
```

In order to understand how this input is interpreted the user has to understand how TART searches for zones. When a particle leaves one zone the code determines which zone it has entered by searching the list of zones and using the bounding surfaces to determine if the current (x, y, z) space point is in a given zone. The list of zones is searched in a circular fashion starting at the most probable zone, if necessary continuing to the maximum zone number defined in the problem, and if necessary continuing from the first zone up to the most probable zone (the entire list of zones is searched in a circular fashion).

Assume that we have a source of particles at the origin of the concentric spheres and we wish to determine the leakage of particles from each sphere. If we wish to determine the leakage from a number of spheres of different radii we could use **jb, zonejb or zone** and run separate calculations for each radius. However, using **bjp** input we can obtain the answers we want in one single calculation.

The trick to use in doing this is to have overlapping zones, where zones of smaller radius are totally contained within zones of larger radius, and zones of smaller radius have smaller zone numbers, and the most probable zone always indicates a zone with a larger radius, with the except of the last, outer most zone; presumably a vacuum non-re-entry zone, where merely for completeness the most probable zone may be defined as the zone just inside it. Lastly it is important when defining the source to indicate that its most probable zone is the inner most zone.

For the above description of zones and a source at the origin all source particles will initially be in zone 1. They will transport until they disappear or cross the first spherical surface and leave zone 1. At this point the search for the zone that has been entered will start at the most probable zone number when leaving zone 1 and crossing surface 1 = zone 2, as indicated by the above input. In this case the search will always indicate that the particle has entered zone 2 (since in this simple example it is the only zone immediately outside of the first spherical surface). Again particles will transport in zone 2 until they disappear or leave the zone. But note that in this case, since the only bounding

surface for zone 2 is surface 2, the outer spherical surface, the particle can never re-enter zone 1; if it crosses the inner spherical surface traveling inward or outward, based on the defined bounding surfaces of zone 2, it remains in zone 2. If the particle does leave zone 2 by crossing the outer spherical surface the search for the next zone will begin at the most probable zone when leaving zone 2 and crossing surface 2 = zone 3, as indicated by the above input. In this case the search will always indicate that the particle has entered zone 3 (since this is the only zone immediately outside the outer spherical surface), and the history will be terminated, assuming that this outer most zone 3 is a vacuum, non-re-entry zone.

So what are our results? If we are scoring what enters each zone, what enters zone 2 is the leakage from a sphere with the inner radius, and what enters zone 3 is the leakage from a sphere with the outer radius. In other words, in one calculation we have calculated the leakage from both spheres.

This is a trivial example, but hopefully the reader can understand that if we had defined ten, twenty, or any number of concentric spheres, the leakage from all of them could be calculated in only one calculation.

Using **bjp** input has a number of advantages and disadvantages.

The advantages include: 1) potentially large savings in computer time if you can calculate many results in a single problem, 2) the results are guaranteed to be statistically correlated, since in the same run you are using exactly the same histories, e.g., if you perform a number of separate calculations due to statistics you may end up with the non-physical conclusion that the leakage across some larger radius spheres is more than that across some smaller radius spheres.

The disadvantages include: 1) this input can be very complicated and it is very easy to make errors and incorrectly define the input, 2) with overlapping geometry it is not possible to use the input checking codes, such as **TARTCHEK**, to help verify the accuracy of your input, 3) there aren't that many real problems where it is possible to completely overlap geometry, as we have done in the simple example problem used above, 4) the savings in computer time isn't as large as one might think; the overall running will be dominated by the largest zone defined.

Based on experience I do not recommend that you use this option. It has been included in TART only for completeness, since it is a TARTNP option. For any real problem involving complicated geometry it is simply far too easy to make mistakes in the input geometry description, that can lead to disastrous results. In my opinion it is better to be conservative and use **jb, zonejb or zone** input with non-overlapping geometry. You can then use the available interactive graphics program **TARTCHEK** to quickly verify your description of geometry and then run TART. Alternatively, if you do decide to use **bjp** and overlapping geometry, you are on your own as far as verifying the accuracy of your description of geometry = caveat emptor.

Reflecting Zones

Using reflecting zones can in some cases greatly simplify input preparation. By taking advantage of any symmetries in a problem you may be able to only model a smaller portion of a problem and use reflection to simulate the entire problem, thus simplifying your input deck and minimizing the amount of work you have to do to prepare the deck. For example, you can simulate an infinitely repeating pattern of planes using only one repetition of the pattern and reflecting boundaries.

TART ALWAYS uses 3-D geometry. Even if you have a 1-D series of planes or an infinite cylinder you MUST close your geometry to define 3-D space finite volume space. For an infinite cylinder you can use two reflecting planes orthogonal to the axis of the cylinder to close your geometry. For a series of planes you can use four reflecting planes orthogonal to your planes to close your geometry.

Be careful when defining reflecting zones to insure that the results really exactly simulate your true geometry; it is very easy to make logic mistakes and end up with a problem description that does not correspond to the geometry you are trying to model. For example, in the case of a series of planes, rather than using four orthogonal planes to close your geometry you might think it is easy and equivalent to use a single reflecting cylinder orthogonal to your planes; this is not equivalent. You can stack an infinite array of cubes to exactly fill your 3-D space by using four reflecting planes and reflection from these planes will exactly simulate what would have happened had a particle actually crossed a plane and entered an adjacent cell. In contrast, you cannot stack a series of cylinders to exactly fill your 3-D space and reflection from the cylindrical surface will not exactly simulate what would have happened had a particle actually crossed the cylinder and entered an adjacent cell.

Warning - when using reflecting boundaries you will obtain the same results as if you had modeled the entire system; this is true only integrated over each zone, but not per unit volume or mass. Remember that the volumes calculated by this code will only be for the real interior zones of your problem, and particles can only transport in these real zones. Therefore per incident source particle calculated total path lengths, etc. will be exactly the same as in a problem where you model the entire system. However, per unit volume the results for these zones will be larger than the result you would obtain not using reflecting zones and modeling the entire system. The results will be larger exactly in proportion of the ratio of the volume of a zone in the entire system to that in your system with reflectors. For example, if you have a cylindrically symmetric system and you use a reflecting plane on its axis to only model one half of the system, the calculated total path length, etc. will be the same as if you had modeled the entire system. But since the volume of your system with a reflector is only half that of the entire system your results per unit volume or mass will be twice as large as it would have been had you modeled the entire system without a reflector. You MUST consider this effect in order to properly interpret the output results.

TART uses specula reflection to reflect neutrons and photons off a boundary when a zone is identified as a reflecting zone. Note, input identifies zones, not surfaces, as reflecting; particles are reflected off of any bounding surface of a zone that is identified as a reflecting zone. For reflection the code calculates the normal to the surface that the particle crossed entering the reflecting zone and derives the direction cosines such that the incident particle, the surface normal, and the reflected particle are coplanar.

For historical reasons TART allows a number of keywords to define that a zone is a reflector. These keywords seem to refer to different types of surfaces, rather than to zones. As actually used in the code all are equivalent; any zone that is identified as reflecting will always reflect particles, regardless of the types of surfaces that bound it, e.g., planes, spheres, cylinders, etc.. The only except is that the code will print the number of particles that have reflected off of **xplane, yplane** or **zplane**. If this information is of interest to you, you should use the appropriate reflection keyword. Otherwise use any of the following keywords interchangeably.

reflx nz1 nz2 nz3...

Zones nz1, nz2, nz3...are reflecting zones. These zones need only one **xplane** boundary. These zones are assigned material -1, to identify them as x reflectors.

refly nz1 nz2 nz3...

Zones nz1, nz2, nz3...are reflecting zones. These zones need only one **yplane** boundary. These zones are assigned material -2, to identify them as y reflectors.

reflz nz1 nz2 nz3...

Zones nz1, nz2, nz3...are reflecting zones. These zones need only one **zplane** boundary. These zones are assigned material -3, to identify them as z reflectors.

reflgp nz1 nz2 nz3...
reflect nz1 nz2 nz3…

Zones nz1, nz2, nz3...are reflecting zones. These zones need only one **genplane** or **gpl** boundary. These zones are assigned material -5, to identify them as general plane reflectors.

reflq nz1 nz2 nz3...

Zones nz1, nz2, nz3...are reflecting zones. These zones need only one quadratic boundary. These zones are assigned material -4, to identify them as quadratic reflectors.

As mentioned above, zones, not surfaces, reflect, so that any of these keywords may be used interchangeable for input. Regardless of what type of surface a particle crosses when entering a zone, the particle will be correctly reflected based on the type of surface

crossed, e.g., plane, sphere, cylinder, etc., regardless of the keyword used on input to define the zone as reflecting. For example, even if you identify a reflecting zone using the keyword **reflx** if the particle crosses a spherical surface to enter the zone it will be correctly reflected from a **sphere**, not a **xplane**.

Restrictions: Reflecting zones MUST not have a material assigned to them. If you assign material to a reflecting zone you can overwrite the assigned negative material numbers, -1 through -5, which is used to define a zone as reflecting (as described above), in which case the zone will no longer be a reflector.

Material Definitions

Each material is defined by specifying its overall density in grams/cc and each one of its constituents. Constituents are defined by specifying either their atomic or weight fraction and the ZA (1000*Z+A) of the isotope or element. Two TART keywords are defined to allow specification by either atomic or weight fraction,

matl m rho f(1) za(1) f(2) za(2).....[&]

m - assigned material number
rho - material overall density in grams/cc
f(1) - atom fraction of the first constituent
za(1) - ZA of the first constituent
f(2) - atom fraction of the second constituent
za(2) - ZA of the second constituent

For example, water has two atoms of hydrogen (ZA=1001) to each atom of oxygen (ZA=8016). For water at an overall density of 1.0 grams/cc the input would be,

```
matl    1   1.0   2.0   1001    1.0   8016
```

Note, the 1 following matl defines this as material number 1.

Restrictions: The overall density MUST be positive. Different material numbers should be assigned to each material. The atom fractions need not be normalized to the actual atoms/cc; these are merely relative atom fractions. Each problem may use ANY number of different materials (the earlier limit of 50 materials no longer applies). Each material may now contain ANY number different constituents (the earlier limit of 25 constituents no longer applies). **matl** and **matlwp** input may both be used in the same problem.

matlwp m rho f(1) za(1) f(2) za(2).....[&]

m - assigned material number
rho - material overall density in grams/cc

f(1) - weight fraction of the first constituent
za(1) - ZA of the first constituent
f(2) - weight fraction of the second constituent
za(2) - ZA of the second constituent

For example, concrete at an overall density of 2.32 grams/cc, with 5% Al rebar and impurities of Ti, Sc and Ni, the input would be,

```
matlwp  2  2.32  .005683  1001  .502588  8016  .026720  11023 &
.017647 12000  .091026  13027  .255033  14000 .062213  20000 &
.0001 24000  .035493  26000  .000199  29000  .000199  30000 &
.001 28000
```

Note, the use of continuation lines, indicated by & as the last character of an input line. Note, the 2 following matl defines this as material number 2.

Restrictions: The overall density MUST be positive. Different material numbers should be assigned to each material. The weight fractions need not be normalized to the actual grams/cc; these are merely relative weight fractions. Each problem may use ANY number of different materials (the earlier limit of 50 materials no longer applies). Each material may now contain ANY number different constituents (the earlier limit of 25 constituents no longer applies).. **matlwp** and **matl** input may both be used in the same problem.

Zonal Properties

You can define the following zonal properties for each zone individually or for all zones; if you do not define the zonal properties by input you will be using the default value indicated in parenthesis,

1) material number assigned to zone (0)
2) minimum neutron energy or zone temperature (2.53e-08 MeV)
3) minimum photon energy (-1.0e-04 MeV)
4) relative density (1.0)
5) neutron weight, used for splitting (1.0)
6) photon weight, used for splitting (1.0)
7) photon production multiplier (1.0)
8) neutron edit (output results) type (3)
9) photon edit (output results) type (3)
10) request additional edits (output results) (0)

Zonal properties are defines in one of two forms,

keyword value nz1 thru nz2

or

keyword value nz1 nz2 nz3.....

These forms cannot be combined on one input line. However, a combination of the two may be used. In any problem latter input lines supersede earlier input lines, so that the most widely used value can first be assigned to all zones and later input lines can change the value for specific zones. For example, if zone 20 has a weight of 1.0, zones 8, 10 and 12 a weight of 2.0, and all remaining zones a weight of 4.0, the following input can be used,

weight 4.0 1 thru 30
weight 2.0 8 10 12
weight 1.0 20

matz m nz1 nz2 nz3.....(0)
matz m nz1 thru nz2

Material number **m** is assigned to the indicated zones, **nz1, nz2, nz3...** or **nz1** through **nz2**. Note, the default (0) in the above example indicates that if you do not explicitly answer a material number, **m**, to a zone, by default material number 0 (void; a terminus zone) is assigned to the zone.

Restrictions: Material number **m** must be defined by **matl** or **matlwp** input. A material number MUST be assigned to all interior zones, A material MUST NOT be assigned to exterior, non-re-retrant zones or reflecting zones.

volume nz1 vz1 nz2 vz2 nz3 vz3…..

The volume (in cc) for zones **nz1, nz2, nz3**… is defined by input to be **vz1, vz2, vz3**…, respectively. TART will attempt to analytically calculate the volume of each zone. When it cannot analytically calculate the volume you can use the input option **mcvdisk or mcvplane** to statistically sample the volume. To accurately statistically sample volumes can be very time consuming. This input option allows you to statistically sample volumes only once, and then in all following calculations you need merely define them by input. This will allow you to correctly define volumes that can effect special output, such as deposition in MeV/cc or MeV/grams, see **probm** and **sentl 35.**

Example Input: The following input line defines the volume of zones 2 17 and 23 to be 1.34, 13.7 and 213.6 cc, respectively,

Volume 2 1.34 17 13.7 23 213.6

Restrictions: The zone numbers **nz1, nz2, nz3**… MUST be 1 to the maximum number of zones used in the problem, and all volumes be **vz1, vz2, vz3**… MUST be positive. The

volumes that you input are used by TART without questioning their accuracy, and no attempt is made to analytically calculate the volumes of these zones.

emin e nz1 nz2 nz3.....(2.53e-08)
emin e nz1 thru nz2 (2.53e-08)

emin has two different meanings depending on whether or not thermal scattering is being used (**sentl 39**). If thermal scattering is not being used **emin** defines the minimum neutron energy in each zone. If thermal scattering is being used **emin** defines the temperature of zones in MeV; in this case only **sentl 8** can be used to define the minimum neutron energy.

When thermal scattering is not being used **emin** and **sentl 8** control the minimum allowed neutron energy (MeV) in each zone. **emin** can be used to define the minimum energy zone by zone; **sentl 8** allows it to be defined globally as the same value for all zones.

When thermal scattering is not being used, the absolute value of **e** is the minimum allowed energy (MeV) of neutrons in these zones. For photons use **eming**. After a collision if a neutron's energy is less than the absolute value of e: 1) if **e** is positive the neutron's energy is set equal to **e**, 2) if **e** is negative the neutron history is terminated.

WARNING – starting with TART2002 the minimum and maximum tracking and tally energies for both neutrons and photons is the minimum and maximum energies read by TART from the binary data files. Currently these limits are: neutrons, 1.0d-11 to 20 MeV, photons, 1.0d-4 to 1000. MeV. **Therefore it is recommended that you never use this option to limit the energy range used in calculations.**

Restrictions: the default minimum energy is 1.0d-11 MeV, and the default temperature is 2.53e-08 (room temperature). The minimum neutron energy MUST be within the limits of the multigroup neutron energy range, currently: 1.0e-11 to 20 MeV. The temperature of a zone must be positive.

eming e nz1 nz2 nz3.....(-1.0e-04)
eming e nz1 thru nz2 (-1.0e-04)

eming and **sentl 9** control the minimum allowed photon energy (MeV) in each zone. **emin** can be used to define the minimum energy zone by zone; **sentl 9** allows it to be defined globally as the same value for all zones.

The absolute value of **e** is the minimum allowed energy (MeV) of photons in these zones. For neutrons use **emin**. After a collision if a photon's energy is less than the absolute value of **e**: 1) if **e** is positive the photon's energy is set equal to **e**, 2) if **e** is negative the photon history is terminated.

WARNING – starting with TART2002 the minimum and maximum tracking and tally energies for both neutrons and photons is the minimum and maximum energies read by

TART from the binary data files. Currently these limits are: neutrons, 1.0d-11 to 20 MeV, photons, 1.0d-4 to 1000. MeV. **Therefore it is recommended that you never use this option to limit the energy range used in calculations.**

Restrictions: the default minimum energy for all zones is -1.0e-04, i.e., terminate any photon history if the photon energy is less than 100 eV. The minimum photon energy MUST be within the limits of the photon data range: 1.0e-04 to 1000 MeV.

eta r nz1 nz2 nz3.....(1.0)

The relative density **r** is assigned to the indicated zones. The overall density of a zone, in grams/cc, is defined as the product of the density of the material assigned to the zone times the relative density.

Restrictions: The default value is 1.0. Relative density must be positive.

weight w nz1 nz2 nz3.... (1.0)
weight w nz1 thru nz2 (1.0)

w is the neutron statistical weight for zone **nz1, nz2, nz3...**or **nz1** through **nz2**, and the photon statistical weight if **wgtgam** input is not used. Weights other than 1.0 can be used in high attenuation problems to force particle flow into spatial regions of interest. At a zone boundary crossing, the product of weight and number of particles is conserved. For example, if a particle crosses from a high weight zone into a low weight zone, the particle is split. If a particle crosses from a low weight zone into a high weight zone, particles are statistically discarded.

Recommendations: **Warning** - Use zonal weights with extreme care. Do not try to drastically change weights over relatively small distances (small in terms of mean free paths). Be absolutely sure that the ratio of weights in adjacent zones is 1/2, 1 or 2; this is often difficult to do in complicated geometries, where one zone may be adjacent to many other zones.

As a rough guideline in using weights, remember that a source will be exponentially attenuated according to the total mean free path. About every 0.7 mean free paths a source will be attenuated by a factor of 2. Therefore if you 1/2 the weight every 0.7 mean free path the number of source particles will remain approximately constant. If you want this to happen you can run a very short calculation (see **sentl 2** and **3** to minimize the number of histories) and reading the output report to determine the mean free path of the materials that you are using at your source energy. This will tell you how to zone your problem to allow you to use weights to propagate a source deep into a medium.

Restrictions: The default weight for all zones is 1.0. Weights are ignored in criticality (static or dynamic) calculations. ALL weights MUST be powers of 2, e.g., 1, 2, 4, 8,...1/2, 1/4, 1/8, etc. The ratio of weights in adjacent zones MUST be 1/2, 1 or 2, i.e., large changes in weights between adjacent zones are not allowed - the weight can change

by at most a factor of 2. In going from a high to low weight zone the particle is split into exactly two particles, and in going from a low to high weight zone one half the particles are discarded - **Warning** - this procedure is correct only if the ratio of weights in adjacent zones is 1/2, 1 or 2.

wgtgam w nz1 nz2 nz3.... (1.0)
wgtgam w nz1 thru nz2 (1.0)

w is the photon statistical weight for zones **nz1, nz2, nz3...**or **nz1** through **nz2**. Weights other than 1.0 can be used in high attenuation problems to force particle flow into spatial regions of interest. At a zone boundary crossing, the product of weight and number of particles is conserved. For example, if a particle crosses from a high weight zone into a low weight zone, the particle is split. If a particle crosses from a low weight zone into a high weight zone, particles are statistically discarded.

Recommendations: **Warning** - Use zonal weights with extreme care. Do not try to drastically change weights over relatively small distances (small in terms of mean free paths). Be absolutely sure that the ratio of weights in adjacent zones is 1/2, 1 or 2; this is often difficult to do in complicated geometries, where one zone may be adjacent to many other zones.

As a rough guideline in using weights, remember that a source will be exponentially attenuated according to the total mean free path. About every 0.7 mean free paths a source will be attenuated by a factor of 2. Therefore if you 1/2 the weight every 0.7 mean free path the number of source particles will remain approximately constant. If you want this to happen you can run a very short calculation (see **sentl 2** and **3** to minimize the number of histories) and reading the output report to determine the mean free path of the materials that you are using at your source energy. This will tell you how to zone your problem to allow you to use weights to propagate a source deep into a medium.

Restrictions: The default weight for all zones is 1.0. **weight** input lines change the default if **wgtgam** input is not used. ALL weights MUST be powers of 2, e.g., 1, 2, 4, 8,...1/2, 1/4, 1/8, etc. The ratio of weights in adjacent zones MUST be 1/2, 1 or 2, i.e., large changes in weights between adjacent zones are not allowed - the weight can change by at most a factor of 2. In going from a high to low weight zone the particle is split into exactly two particles, and in going from a low to high weight zone one half the particles are discarded - **Warning** - this procedure is correct only if the ratio of weights in adjacent zones is 1/2, 1 or 2.

gpwgt w nz1 nz2 nz3.... (1.0)
gpwgt w nz1 thru nz2 (1.0)

Photon (gamma ray) production multiplier. **w** multiplies the number of photon produced per neutron collision. The resulting photons are tracked and scored with weight **1/w**.

Recommendations: photon multipliers can be used to good advantage in combination with neutron (**weight**) and photon (**wgtgam**) statistical weights to minimize variance and speed convergence, particularly in thick media that strongly attenuate photons, e.g., almost any high atomic number (Z) material. In this case without using photon multipliers you may not generate enough photons deep enough in the material to obtain statistically reliable results.

Restrictions: The default is 1.0, i.e., track and score exactly the number of photons produced per neutron collision. The multiplier MUST be positive. **Warning** - you CANNOT use this option to try and turn off photon production and tracking in zones by using a multiplier of zero, since photons are assigned a weight of **1/w**.

For zone by zone output tally options for neutrons (**ltype**) and photons (**ltypeg**), as well as **notal**, **units** and **zonemult** see **Neutron and Photon Edit and Tally (Output) types**, later in this report.

Sources

Every problem MUST include a source; this includes reactivity calculations, as well as neutron and/or photon source problems.

In order to uniquely define a source distribution you MUST define the source: 1) type of particle (neutrons and/or photons), 2) energy spectrum, 3) angular distribution, 4) time distribution, and 5) spatial distribution. All source parameters have default values.

Type of Particle

sentl 1 defines the type of source particles and what particles to track. The default value is **sentl 1 0**, neutron source particles, track neutrons and neutron induced photon production. **sentl 1** can be used to define other types of sources, see **sentl 1**, below.

Energy Distribution

Neutron spectrum
The default is a neutron induced fission spectrum of neutrons. For monoenergetic sources the energy can be defined using **sentl 4**. For an actual spectrum there are many options, that are described below.

Photon spectrum
The default is a neutron induced fission spectrum of photons. For monoenergetic sources the energy can be defined using **sentl 17**. For an actual spectrum there are many options, that are described below.

Neutrons	Photons	Distribution
4	**17**	sentinel to define monoenergetic source
34		sentinel for fission spectrum temperature

ediscr	**ediscrg**	a series of monoenergetic sources
s9or10e	**s9or10eg**	energy dependent spectrum
espec	**especg**	energy dependent spectrum
fspec	**fspecg**	energy dependent spectrum
ehist	**ehistg**	histogram energy dependent spectrum
eangl	**eanglg**	correlated angle-energy distribution
enerangl	**enranglg**	correlated angle-energy distribution
maee	**maeeg**	correlated angle-energy distribution
maeeh	**maeehg**	correlated angle-energy distribution
maec	**maecg**	correlated angle-energy-time distribution
	blkbdy	blackbody photon source spectrum
ebias	**ebiasg**	bias energy distribution
maxwell		**Maxwellian neutron source spectrum (TART2002)**

Each of these is described in detail below.

blkbdy [temp emax thin]

The photon source will be a blackbody spectrum with a temperature of **temp** extending up to a maximum energy of **emax**, with a maximum fractional deviation of **thin** from an exact blackbody spectrum.

Restrictions: **temp, emax** and **thin** are optional input that default to **temp** = 0.001 MeV, **emax** = 15.0***temp** and **thin** = 0.003. If input **temp, emax** and **thin** MUST all be positive.

TART2002 Update

maxwell temp

The neutron source will be a Maxwellian spectrum with a temperature of **temp**, in MeV. The Maxwellian will extend down to **temp**/5000 and up to 8***temp**, constrained to be within the energy range of the nuclear data being used, typically 1.0e-11 to 20 MeV.

For example,

maxwell 2.53d-8

Defines a room temperature thermal energy Maxwellian with a temperature of 0.0253 eV.

maxwell 1.89

Defines a fission spectrum Maxwellian with a temperature of 1.89 MeV.

Restrictions: The temperature, **temp**, MUST be positive. The energy range of the spectrum is constrained to be within the energy range of the nuclear data being used, typically 1.0e-11 to 20 MeV.

ebias j w(j) e(j) w(j+1) e(j+1).....

The neutron source spectrum is biased by assigning weights using the piece wise linear distribution defined by the weights **w(j)** and energies **e(j)**.

Restrictions: The minimum and maximum energies MUST correspond EXACTLY to those of the input spectrum. The number of neutrons sampled at each energy will be determined by the sampling weight, and the relative weight of each sampled neutron will be equal to the ratio of the actual spectrum to the weighted spectrum.

ebiasg j w(j) e(j) w(j+1) e(j+1).....

The photon source spectrum is biased by assigning weights using the piece wise linear distribution defined by the weights **w(j)** and energies **e(j)**.

Restrictions: The minimum and maximum energies MUST correspond EXACTLY to those of the input spectrum. The number of photons sampled at each energy will be determined by the sampling weight, and the relative weight of each sampled photon will be equal to the ratio of the actual spectrum to the weighted spectrum.

ediscr fn1 e1 fn2 e2.....

The neutron source spectrum will have discrete energies, with the relative number of neutrons **fn1, fn2....** at the discrete energies **e1, e2,.....**.

Restrictions: Do not use for reactivity problems. Any number of **ediscr** input lines may be used. All **ediscr** input lines are combined to define the final relative number of neutrons at each discrete energy.

ediscrg fn1 e1 fn2 e2.....

The photon source spectrum will have discrete energies, with the relative number of photons **fn1, fn2....** at the discrete energies **e1, e2,.....**.

Restrictions: Any number of **ediscrg** input lines may be used. All **ediscrg** input lines are combined to define the final relative number of photons at each discrete energy.

ehist j rn(j) e(j) rn(j+1) e(j+1)..... 0.0 emax

The neutron source spectrum is a histogram distribution defined by the relative number of neutrons per unit energy **rn(j)** in the energy interval **e(j)** to **e(j+1)**.

Restrictions: Any number of **ehist** input lines may be used to define the complete spectrum. The final **rn** MUST be 0.0 at the maximum energy **emax**. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: The following input defines a histogram spectrum between 6.7743 and 15.242 MeV,

```
ehist    1  0.01433  6.7743  0.01010  7.5479  0.007873  8.3633
ehist    4  0.004204  9.2205  0.006741  10.120  0.002353  11.060
ehist    7  0.003448  11.547  0.003121  12.043  0.00434  12.550
ehist   10  0.02450  13.068  0.02234  13.863  0.02609  14.134
ehist   13  0.01429  14.407  0.0  15.242
```

ehistg j rn(j) e(j) rn(j+1) e(j+1)..... 0.0 emax

The photon source spectrum is a histogram distribution defined by the relative number of photons per unit energy **rn(j)** in the energy interval **e(j)** to **e(j+1)**.

Restrictions: Any number of **ehistg** input lines may be used to define the complete spectrum. The final **rn** MUST be 0.0 at the maximum energy **emax**. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: See, **ehist**, above for example input.

eangl and **eanglg** are used to define the angular distribution of neutron and photon sources, and are used in conjunction with **enerangl** and **enranglg** to define a completely correlated angle-energy distribution. For a description of **eangl** and **eanglg** see **ANGULAR DISTRIBUTIONS**.

enerangl j e(j) c(j) e(j+1) c(j+1).....

The neutron source spectrum is a completely correlated angle-energy distribution. **eangl** defines the angular distribution and is first sampled to define a cosine. **enerangl** defines the completely correlated energy, in terms of a piece wise linearly interpolable table of energies, **e(j)**, corresponding to any sampled cosine **c(j)**.

Restrictions: The tabulated cosine range defined by the **enerangl c(j)** input MUST span at least the same range as the cosine range defined by **eangl** input.

enranglg j e(j) c(j) e(j+1) c(j+1).....

The photon source spectrum is a completely correlated angle-energy distribution. **eanglg** defines the angular distribution and is first sampled to define a cosine. **enranglg** defines the completely correlated energy, in terms of a piece wise linearly interpolable table of energies, **e(j)**, corresponding to any sampled cosine **c(j)**.

Restrictions: The tabulated cosine range defined by the **enranglg c(j)** input MUST span at least the same range as the cosine range defined by **eanglg** input.

espec j rn(j) e(j) rn(j+1) e(j+1).....

The neutron source spectrum is a piecewise linearly interpolable defined by the relative number of neutrons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Any number of **espec** input lines may be used to define the entire spectrum. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: The following input defines a piecewise linear spectrum between 6.7743 and 15.242 MeV. Note, the below spectrum is similar to the spectrum defined above for **ehist**, however here when we use the keyword **espec** the spectrum is interpreted as piecewise linear, whereas in the case of **ehist** it is interpreted as histograms,

```
espec   1  0.01433  6.7743  0.01010  7.5479  0.007873  8.3633
espec   4  0.004204  9.2205  0.006741  10.120  0.002353  11.060
espec   7  0.003448  11.547  0.003121  12.043  0.00434  12.550
espec 10  0.02450  13.068  0.02234  13.863  0.02609  14.134
espec 13  0.01429  14.407  0.0  15.242
```

especg j rn(j) e(j) rn(j+1) e(j+1).....

The photon source spectrum is a piecewise linearly interpolable defined by the relative number of photons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Any number of **especg** input lines may be used to define the entire spectrum. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: See, **espec**, above for example input.

fspec j f(j) e(j) f(j+1) e(j+1).....

The neutron source spectrum is a piecewise linearly interpolable defined by the relative number of neutrons per unit energy **rn(j)** at each energy **e(j)**, where **rn(j) = f(j)/e(j)**.

Restrictions: Any number of **fspec** input lines may be used to define the entire spectrum. All energies **e(j)** MUST be positive; otherwise in attempting to define **rn(j) = f(j)/e(j)** there will be an obvious problem. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: The following input defines a piecewise linear spectrum between 6.7743 and 15.242 MeV. Note, the below spectrum is similar to the spectrum defined above for **ehist**, and **espec**, however here when we use the keyword **fspec** the spectrum is

interpreted as piece wise linear, but with a variation that is different from the **espec** interpretation, since here the relative number of neutrons is **rn(j) = f(j)/e(j)**,

```
fspec   1  0.01433  6.7743  0.01010  7.5479  0.007873  8.3633
fspec   4  0.004204  9.2205  0.006741  10.120  0.002353  11.060
fspec   7  0.003448  11.547  0.003121  12.043  0.00434  12.550
fspec 10  0.02450  13.068  0.02234  13.863  0.02609  14.134
fspec 13  0.01429  14.407  0.0  15.242
```

fspecg j f(j) e(j) f(j+1) e(j+1).....

The photon source spectrum is a piecewise linearly interpolable defined by the relative number of photons per unit energy **rn(j)** at each energy **e(j)**, where **rn(j) = f(j)/e(j)**.

Restrictions: Any number of **fspecg** input lines may be used to define the entire spectrum. All energies **e(j)** MUST be positive; otherwise in attempting to define **rn(j) = f(j)/e(j)** there will be an obvious problem. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: See, **fspec**, above for example input.

For a description of **maec** and **maecg**, that are used by define angular and time distributions to be used in conjunction with **maee**, **maeeh**, **maeeg** and **maeehg**, to define correlated angle-energy-time distributions, see **Angular Distributions.**

maee i j rn(j) e(j) rn(j+1) e(j+1).....

Neutron source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maec** input and the energy distribution is defined by **maee** input. The neutron source spectrum is a piecewise linearly interpolable defined by the relative number of neutrons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Do not use for reactivity problems. Any number of **maee** input lines may be used to define the entire spectrum. The energies **e(j)** MUST be monotonically increasing or decreasing. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: The following input uses **maec** and **maee** input to define two different spectra: the first for the cosine range 1.0 to 0.9, and the second for the cosine range 0.9 to 0.866. Note, the **maec** input defines the angular ranges (with a default launch time = 0) and the **maee** input defines the spectrum in each cosine range. Note, since **maee** is used the input is interpreted as piece wise linearly interpolable, in contrast to **maeeh** input which is interpreted as histogram.

```
* Neutron source for cosine = 1.0 to 0.9
maec   1     167     1.0   0.9
```

```
maee   1      1     3.08e-07   3.316e-02  9.26e-07   6.74e-02
maee   1      3     5.86e-06   9.6e-02    2.56e-05   0.135
maee   1      5     8.27e-05   0.318      2.4e0-4    0.75
maee   1      7     2.19e-04   2.02
* Neutron source for cosine = 0.9 to 0.866
maec   2    753     0.9   0.866
maee   2      1     6.17e-07   4.73e-02   4.6e-06    7.8e-02
maee   2      3     1.04e-07    0.1        1.99e-04   0.18
maee   2      5      6.2e-04    0.42       1.01e-03   1.0
maee   2      7      5.6e-04    2.8
```

maeeg i j rn(j) e(j) rn(j+1) e(j+1).....

Photon source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maecg** input and the energy distribution is defined by **maeeg** input. The neutron source spectrum is a piecewise linearly interpolable defined by the relative number of photons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Any number of **maeeg** input lines may be used to define the entire spectrum. The energies **e(j)** MUST be monotonically increasing or decreasing. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example input: see above under **maee**.

maeeh i j rn(j) e(j) rn(j+1) e(j+1).....0.0 emax

Neutron source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maec** input and the energy distribution is defined by **maeeh** input. The neutron source spectrum is a histogram distribution defined by the relative number of neutrons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Do not use for reactivity problems. Any number of **maee** input lines may be used to define the entire spectrum. The energies **e(j)** MUST be monotonically increasing or decreasing. The final **rn** MUST be 0.0 at the maximum energy **emax**. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example Input: The following input uses **maec** and **maeeh** input to define two different spectra: the first for the cosine range 1.0 to 0.9, and the second for the cosine range 0.9 to 0.866. Note, the **maec** input defines the angular ranges (with a default launch time = 0) and the **maeeh** input defines the spectrum in each cosine range. Note, since **maeeh** is used the input is interpreted as a histogram distribution, in contrast to **maee** input which is interpreted as piece wise linearly.

```
* Neutron source for cosine = 1.0 to 0.9
maec   1    167     1.0   0.9
maeeh  1      1     3.08e-07   3.316e-02  9.26e-07   6.74e-02
```

```
maeeh 1     3    5.86e-06  9.6e-02    2.56e-05  0.135
maeeh 1     5    8.27e-05  0.318      2.4e0-4   0.75
maeeh 1     7    0.0       2.02
* Neutron source for cosine = 0.9 to 0.866
maec  2   753    0.9  0.866
maeeh 2     1    6.17e-07  4.73e-02   4.6e-06   7.8e-02
maeeh 2     3    1.04e-07   0.1        1.99e-04  0.18
maeeh 2     5     6.2e-04   0.42       1.01e-03  1.0
maeeh 2     7     0.0       2.8
```

maeehg i j rn(j) e(j) rn(j+1) e(j+1).....0.0 emax

Photon source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maecg** input and the energy distribution is defined by **maeehg** input. The neutron source spectrum is a histogram distribution defined by the relative number of photons per unit energy **rn(j)** at each energy **e(j)**.

Restrictions: Any number of **maeehg** input lines may be used to define the entire spectrum. The energies **rn(j)** MUST be monotonically increasing or decreasing. The final **rn** MUST be 0.0 at the maximum energy **emax**. The code integrates the entire spectrum to define 1000 equally probable sampling bins.

Example input: see above under **maeeh**.

s9or10e i j rn(j) e(j) rn(j+1) e(j+1).....

For a neutron shell source defined the **i-th source9** or **source10** input line, the initial source energy distribution will be sampled from a number of discrete energies, where **rn(j)** is the relative number of neutrons at discrete energies **e(j)**.

Restrictions: All energies MUST be within the range of the neutron data, 1.0e-11 to 20 MeV for 616 groups. Do not use with reactivity problems.

s9or10eg i j rn(j) e(j) rn(j+1) e(j+1).....

For a photon shell source defined the **i-th s9g** or **s10g** input line, the initial source energy distribution will be sampled from a number of discrete energies, where **rn(j)** is the relative number of photons at discrete energies **e(j)**.

Restrictions: All energies MUST be within the range of the photon data, 1.0e-04 to 30 MeV.

Angular Distribution

Neutron Angular Distribution

The default is an isotropic angular distribution. The angular distribution can be defined using **sentl 6** and **7**, or using one of the options described below.

Photon Angular Distribution
The default is an isotropic angular distribution. The angular distribution can be defined using **sentl 41** and **42**, or using one of the options described below.

Neutrons	Photons	Distribution
anglsrce	**anglsrg**	define angular range
6, 7	**41, 42**	sentinels to define angular range
axis	**axisg**	switch source orientation
eangl	**eangl g**	correlated angle-energy distribution
enerangl	**enranglg**	correlated angle-energy distribution
maee	**maeeg**	correlated angle-energy distribution
maeeh	**maeehg**	correlated angle-energy distribution
maec	**maecg**	correlated angle-energy-time distribution

anglsrce a1 a2

The neutron source angular distribution will be uniform between the angles **a1** and **a2** (degrees) from the reference axis, which by default is the positive z axis.

Restrictions: Only one **anglsrce** input can be used in a problem. **anglsrce** is an alternative to **sentl 6** and **7** input to define the neutron source angular distribution. The reference axis can be changed using **sentl 32, axis, source3, source4** or **source6**.

anglsrg a1 a2

The photon source angular distribution will be uniform between the angles **a1** and **a2** (degrees) from the reference axis, which by default is the positive z axis.

Restrictions: Only one **anglsrg** input can be used in a problem. **anglsrce** is an alternative to **sentl 41** and **42** input to define the neutron source angular distribution. The reference axis can be changed using **sentl 45, axisg, s3g, s4g** or **s6g**.

axis x1 y1 z1 x2 y2 z2

The neutron source reference axis for launching **source3, source4** or **source9** is the directed line between the two points (**x1, y1, z1**) and (**x2, y2, z2**).

Restrictions: (**x1, y1, z1**) and (**x2, y2, z2**) MUST be two distinct points. **sentl 30** (switch reference axis) is ignored. The angular distribution about the reference axis may be defined using **anglsrce** or **sentl 6** and **7**.

axisg x1 y1 z1 x2 y2 z2

The photon source reference axis for launching **s3g, s4g** or **s9g** is the directed line between the two points (**x1, y1, z1**) and (**x2, y2, z2**).

Restrictions: (**x1, y1, z1**) and (**x2, y2, z2**) MUST be two distinct points. **sentl 32** (switch reference axis) is ignored. The angular distribution about the reference axis may be defined using **anglsrg** or **sentl 41** and **42**.

eangl j rn(j) c(j) rn(j+1) c(j+1)....

The neutron source angular distribution will be sampled from the piecewise linear distribution defined by the relative number of neutrons **rn(j)** and the cosines **c(j)**. The neutron energy will then be defined by interpolation in the table of energy versus direction cosine define by **enerangl** input.

Restrictions: The reference axis for the cosines **c(j)** is the x axis. When either is used both **eangl** and **enerangl** MUST be defined by input.

eanglg j rn(j) c(j) rn(j+1) c(j+1)....

The photon source angular distribution will be sampled from the piecewise linear distribution defined by the relative number of neutrons **rn(j)** and the cosines **c(j)**. The photon energy will then be defined by interpolation in the table of energy versus direction cosine define by **enranglg** input.

Restrictions: The reference axis for the cosines **c(j)** is the z axis. When either is used both **eanglg** and **enranglg** MUST be defined by input.

maec i w c1 c2 [t1 t2]

Neutron source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maec** input and the energy distribution is defined by **maee** or **maeeh** input. For **maec** input the relative number of neutrons to launch for source spectrum **i** is defined by weight **w**. The neutrons are launched in a uniform distribution between the cosines **c1** and **c2** and between times **t1** and **t2**.

Restrictions: Do not use for reactivity problems. **t1** and **t2** are optional input that default to 0, 0. Any number of **maec** input lines may be used to completely define the neutron source distribution. The source number **i** is used in both **maec** and **maee** or **maeeh** input to define which input to correlate. The reference axis is the z axis, unless specified by a **source6** input line, or **source13** input (which uses the normal to a spherical surface).

For a description of **maee** and **maeeh** input to use with **maee** input to define a correlated angle-energy-time neutron source distribution, see **Energy Distribution**.

maecg i w c1 c2 [t1 t2]

Photon source energy spectrum **i** is a correlated angle-energy-time distribution. The angular-time distribution is defined by **maecg** input and the energy distribution is defined by **maeeg** or **maeehg** input. For **maecg** input the relative number of photons to launch for source spectrum **i** is defined by weight **w**. The photons are launched in a uniform distribution between the cosines **c1** and **c2** and between times **t1** and **t2**.

Restrictions: **t1** and **t2** are optional input that default to 0, 0. Any number of **maecg** input lines may be used to completely define the photon source distribution. The source number **i** is used in both **maecg** and **maeeg** or **maeehg** input to define which input to correlate. The reference axis is the z axis, unless specified by a **s6g** input line, or **s13g** input (which uses the normal to a spherical surface).

For a description of **maeeg** and **maeehg** input to use with **maeeg** input to define a correlated angle-energy-time photon source distribution, see **Energy Distribution**.

Time Distribution

Neutron Time distribution
The default is time independent or initial starting time = 0. The initial time for neutron sources can be defined using **sentl 31**. For an actual time distribution use one of the options described below.

Photon Time distribution
The default is time independent or initial starting time = 0. The initial time for photon sources can be defined using **sentl 44**. For an actual time distribution use one of the options described below.

Neutrons	**Photons**	**Distribution**
31	**44**	sentinels to define initial time
timdist	**timdistg**	time distribution
timspec	**timspecg**	time distribution
maec	**maecg**	correlated angle-energy-time distribution

For a description of **maec** and **maecg**, that are used by define angular and time distributions to be used in conjunction with **maee**, **maeeh**, **maeeg** and **maeehg**, to define correlated angle-energy-time distributions, see **Angular Distribution.** For a description of **maee**, **maeeh**, **maeeg** and **maeehg**, see **Energy Distribution.**

timdist j t(j) t(j+1) t(j+2).....

The neutron source's time coordinate will be sampled from the equally probability time intervals **t(j), t(j+1), t(j+2)....**(shakes), with uniform sampling in each time interval.

Restrictions: Do not use in reactivity problems. The times **t(j)** must be positive and in increasing input order. At least one pair of times **t(j), t(j+1)** MUST be defined to allow sampling over some time interval.

timdistg j t(j) t(j+1) t(j+2).....

The photon source's time coordinate will be sampled from the equally probability time intervals **t(j), t(j+1), t(j+2)....**(shakes), with uniform sampling in each time interval.

Restrictions: The times **t(j)** must be positive and in increasing input order. At least one pair of times **t(j), t(j+1)** MUST be defined to allow sampling over some time interval.

timspec j rn(j) t(j) rn(j+1) t(j+1).....

The neutron source's time coordinate will be sampled from the continuous, piecewise linear spectrum defined by the relative number of particles **rn(j)** at each time **t(j).**

Restrictions: Do not use in reactivity problems. At least one pair of times **t(j), t(j+1)** MUST be defined to allow sampling over some time interval.

timspecg j rn(j) t(j) rn(j+1) t(j+1).....

The photon source's time coordinate will be sampled from the continuous, piecewise linear spectrum defined by the relative number of particles **rn(j)** at each time **t(j).**

Restrictions: At least one pair of times **t(j), t(j+1)** MUST be defined to allow sampling over some time interval.

Spatial Distribution

The default spatial distribution is a point source at the origin of the coordinate system. The options are,

Neutrons	Photons	Distribution
source1	**s1g**	a point source
source2	**s2g**	cylindrical shell source
source3	**s3g**	spherical shell source
source4	**s4g**	spherical surface source
source5	**s5g**	rectangular parallelpiped source
source6	**s6g**	directed point source
source9	**s9g**	multiple spherical shell sources
s9		"
source10	**s10g**	multiple cylindrical shell sources
s10		"
s11cone	**s11coneg**	multiple conical surface sources
s11cyl	**s11cylg**	multiple cylindrical surface sources
s11dsk	**s11dskg**	multiple annulus of disk sources
s11sph	**s11sphg**	multiple spherical surface sources
source12	**s12g**	general source read from a file

s12iso		isotropic general source read from a file
source13	**s13g**	segment of a spherical surface source
source15	**s15g**	segment of a cylindrical surface source
s15		"
source16	**s16g**	arc of a spherical surface source
source17	**s17g**	directed spherical surface source
32	**45**	sentinels to switch source axii
source19	**s19g**	spherical shell source in zone
source20	**s20g**	cylindrical shell source in zone
source21	**s21g**	rectangular parallelpiped in zone

Each source spatial distribution includes: 1) a TART keyword defining the type of source, 2) the most probable zone that the source is in, 3) the parameters defining the source coordinates.

The most probable zone is used to optimize the search for the starting zone of each source particle. In many cases the entire spatial source distribution will be in a single zone, in which case it is easy to define the most probable zone. In other cases the distribution may be in a number of zones, in which case you can define the most probable zone as the zone where most of the source particle will originate. In complicated geometries with complicated source distributions it may not be possible for you to decide what is the most probable zone. In this case, don't worry about it. Remember specifying the most probable zone is used merely to optimize the search for the starting zone. If you are not sure what zone this is, just guess, and specify any reasonable zone number that is being used in your problem. The worst a bad guess can do is cause the code to run for a slightly longer time.

The restriction on all sources is that they be within the real interior zones. They cannot lie outside the geometry, nor should point sources be on the boundary of one or more zones. If you have a defined geometry and want to see the effect of particles incident on your geometry you can either: 1) define an additional zone containing air and adjacent to your geometry, so that you can start your source in this zone, or, 2) start you source very close to, but inside the surface of a zone, e.g., for a source incident on a series of planes, start the source 1.0e-06 cm inside the plane closest to the source.

source1 nz [x0 y0 z0]

A neutron point source at (**x0, y0, z0**). The most probable zone that the source is located in is nz.

Restrictions: **x0, y0, z0** are optional input that default to 0, 0, 0 (the origin) if not input. The point (**x0, y0, z0**) cannot be on the boundary of any zone. To avoid a point source on a boundary move the source slightly (1.0e-08 cm) off the boundary and into a real interior zone. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 32**.

s1g nz [x0 y0 z0]

A photon point source at (**x0, y0, z0**). The most probable zone that the source is located in is nz.

Restrictions: **x0, y0, z0** are optional input that default to 0, 0, 0 (the origin) if not input. The point (**x0, y0, z0**) cannot be on the boundary of any zone. To avoid a point source on a boundary move the source slightly (1.0e-08 cm) off the boundary and into a real interior zone. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 45**.

source2 nz ri ro [z0 zlen x0 y0]

A neutron source uniformly distributed in a right circular cylindrical shell parallel to the **z** axis, with inner radius **ri**, and outer radius **ro**. The center of the base of the cylinder is at (**x0, y0, z0**) and its length along the **z** axis is **zlen**.

Restrictions: **z0, zlen, x0, y0** are option input that default to 0, 0, 0, 0. If **zlen** = 0, the source is an annular disk. To align the cylinder with the x or y axis use **sentl 30**. To define a source in an angular sector of a cylinder use **source15**. The reference axis for launching particles is the z axis; the reference axis can be changed using **sentl 32**.

s2g nz ri ro [z0 zlen x0 y0]

A photon source uniformly distributed in a right circular cylindrical shell parallel to the **z** axis, with inner radius **ri**, and outer radius **ro**. The center of the base of the cylinder is at (**x0, y0, z0**) and its length along the **z** axis is **zlen**.

Restrictions: **z0, zlen, x0, y0** are option input that default to 0, 0, 0, 0. If **zlen** = 0, the source is an annular disk. To align the cylinder with the x or y axis use **sentl 43**. To define a source in an angular sector of a cylinder use **s15g**. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 45**.

source3 nz ri ro [z0 x0 y0 a1 a2 dcx dcy dcz b1 b2]

A neutron source uniformly distributed over a segment of a spherical shell with inner radius **ri** and output radius **ro**, centered at the point (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the axis of symmetry defined by **dcx, dcy, dcz**. When the axis of symmetry is the **z** axis, an angular section may also be specified between the angles **b1** and **b2**, which is measured clockwise from the y axis toward the x axis.

Restrictions: **z0, x0, y0** are option input that default to 0, 0, 0 (the origin). **a1, a2** are optional input that default to 180, 0 degrees. **dcx, dcy, dcz** are optional input that default to 0, 0, 1 (the axis of symmetry is the z axis). **b1, b2** are optional input that default to 360, 0 degrees. **b1** and **b2** are used only if the axis of symmetry is the z axis. The

reference axis for launching particles is the Z axis; the reference axis can be changed using an **axis** input line.

s3g nz ri ro [z0 x0 y0 a1 a2 dcx dcy dcz b1 b2]

A photon source uniformly distributed over a segment of a spherical shell with inner radius **ri** and output radius **ro**, centered at the point (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the axis of symmetry defined by **dcx, dcy, dcz**. When the axis of symmetry is the **z** axis, an angular section may also be specified between the angles **b1** and **b2**, which is measured clockwise from the y axis toward the x axis.

Restrictions: **z0, x0, y0** are option input that default to 0, 0, 0 (the origin). **a1, a2** are optional input that default to 180, 0 degrees. **dcx, dcy, dcz** are optional input that default to 0, 0, 1 (the axis of symmetry is the z axis). **b1, b2** are optional input that default to 360, 0 degrees. **b1** and **b2** are used only if the axis of symmetry is the z axis. The reference axis for launching particles is the z axis; the reference axis can be changed using an **axisg** input line.

source4 nz r [z0 x0 y0 a1 a2 dcx dcy dcz b1 b2]

A neutron source uniformly distributed over a segment of a spherical surface of radius **r**, centered at the point (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the axis of symmetry defined by **dcx, dcy, dcz**. When the axis of symmetry is the z axis, an angular section may also be specified between the angles **b1** and **b2**, which is measured clockwise from the y axis toward the x axis.

Restrictions: **z0, x0, y0** are option input that default to 0, 0, 0 (the origin). **a1, a2** are optional input that default to 180, 0 degrees. **dcx, dcy, dcz** are optional input that default to 0, 0, 1 (the axis of symmetry is the z axis). **b1, b2** are optional input that default to 360, 0 degrees. **b1** and **b2** are used only if the axis of symmetry is the z axis. The reference axis for launching particles is the z axis; the reference axis can be changed using an **axis** input line.

s4g nz r [z0 x0 y0 a1 a2 dcx dcy dcz b1 b2]

A photon source uniformly distributed over a segment of a spherical surface of radius **r**, centered at the point (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the axis of symmetry defined by **dcx, dcy, dcz**. When the axis of symmetry is the z axis, an angular section may also be specified between the angles **b1** and **b2**, which is measured clockwise from the y axis toward the x axis.

Restrictions: **z0, x0, y0** are option input that default to 0, 0, 0 (the origin). **a1, a2** are optional input that default to 180, 0 degrees. **dcx, dcy, dcz** are optional input that default to 0, 0, 1 (the axis of symmetry is the z axis). **b1, b2** are optional input that default to 360, 0 degrees. **b1** and **b2** are used only if the axis of symmetry is the z axis. The

reference axis for launching particles is the Z axis; the reference axis can be changed using an **axisg** input line.

source5 nz [z0 zlen x0 xlen y0 ylen]

A neutron source uniformly distributed over a rectangular parallelepiped aligned with the axes and bounding by the planes **x0, x0 + xlen, y0, y0 + ylen, z0, z0 + zlen**.

Restrictions: **z0, zlen, x0, xlen, y0, ylen** are all optional input that default to 0. If any of the lengths are zero, the source region may become a plane, a line, or a point. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 32**.

s5g nz [z0 zlen x0 xlen y0 ylen]

A photon source uniformly distributed over a rectangular parallelepiped aligned with the axes and bounding by the planes **x0, x0 + xlen, y0, y0 + ylen, z0, z0 + zlen**.

Restrictions: **z0, zlen, x0, xlen, y0, ylen** are all optional input that default to 0. If any of the lengths are zero, the source region may become a plane, a line, or a point. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 45**.

source6 nz x1 y1 z1 x2 y2 z2

A neutron point source at (**x1, y1, z1**). The reference axis for launching particles is the direction from the source point (**x1, y1, z1**), toward the point (**x2, y2, z2**).

Restrictions: (**x1, y1, z1**) cannot be the same point as (**x2, y2, z2**). The angular distribution about the reference axis may be specified either by **sentl 6** and **7**, **anglsrce**, or by **maec**. **maec** and either **maee** or **maeeh** input may be used with **source6**.

s6g nz x1 y1 z1 x2 y2 z2

A photon point source at (**x1, y1, z1**). The reference axis for launching particles is the direction from the source point (**x1, y1, z1**), toward the point (**x2, y2, z2**).

Restrictions: (**x1, y1, z1**) cannot be the same point as (**x2, y2, z2**). The angular distribution about the reference axis may be specified either by **sentl 41** and **42**, **anglsrg**, or by **maecg**. **maecg** and either **maeeg** or **maeehg** input may be used with **source6**.

source9 nz w ri ro [z0 x0 y0 t]
s9 nz w ri ro [z0 x0 y0 t]

A neutron source of multiple spherical shells, with each spherical shell defined by a separate **source9** input. The neutron source for each spherical shell has a relative number

of neutrons **w** and is uniformly distributed in a spherical shell of inner radius **ri** and outer radius **ro**, centered at (**x0, y0, z0**). The initial time when neutrons are launched is **t**.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0 (the origin). **t** is optional input that defaults to 0. The reference axis for launching particles is the Z axis; the reference axis can be changed using an **axis** input line. Each spherical shell is assigned an index, 1, 2, 3, etc.,, in the order of **source9** input, that can be used to reference a shell when using **s9or10e** input.

s9g nz w ri ro [z0 x0 y0 t]

A photon source of multiple spherical shells, with each spherical shell defined by a separate **s9g** input. The photon source for each spherical shell has a relative number of photons **w** and is uniformly distributed in a spherical shell of inner radius **ri** and outer radius **ro**, centered at (**x0, y0, z0**). The initial time when photons are launched is **t**.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0 (the origin). **t** is optional input that defaults to 0. The reference axis for launching particles is the Z axis; the reference axis can be changed using an **axisg** input line. Each spherical shell is assigned an index, 1, 2, 3, etc.,, in the order of **s9g** input, that can be used to reference a shell when using **s9or10eg** input.

source10 nz w ri ro [z0 zlen x0 y0 t]
s10 nz w ri ro [z0 zlen x0 y0 t]

A neutron source of multiple cylindrical shells aligned with the z axis, with each cylindrical shell defined by a separate **source10** input. The neutron source for each cylindrical shell has a relative number of neutrons **w** and is uniformly distributed in a cylindrical shell of inner radius **ri** and outer radius **ro**. One end of the cylinder is centered at (**x0, y0, z0**), and the length of the cylinder is **zlen** along the positive z axis. The initial time when neutrons are launched is **t**.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0 (the origin). **zlen** is option input that defaults to 0, in which case the source is an annular disk. **t** is optional input that defaults to 0. To align the cylinder with the x or y axis use **sentl 30**. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 32** input line. Each cylindrical shell is assigned an index, 1, 2, 3, etc.,, in the order of **source10** input, that can be used to reference a shell when using **s9or10e** input.

s10g nz w ri ro [z0 zlen x0 y0 t]

A photon source of multiple cylindrical shells aligned with the z axis, with each cylindrical shell defined by a separate **s10g** input. The photon source for each cylindrical shell has a relative number of photons **w** and is uniformly distributed in a cylindrical shell of inner radius **ri** and outer radius **ro**. One end of the cylinder is centered at (**x0, y0, z0**),

and the length of the cylinder is **zlen** along the positive z axis. The initial time when photons are launched is **t**.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0 (the origin). **zlen** is option input that defaults to 0, in which case the source is an annular disk. **t** is optional input that defaults to 0. To Align the cylinder with the x or y axis use **sentl 43**. The reference axis for launching particles is the Z axis; the reference axis can be changed using **sentl 45** input line. Each cylindrical shell is assigned an index, 1, 2, 3, etc.,, in the order of **s10g** input, that can be used to reference a shell when using **s9or10eg** input.

s11cone nz w z0 z1 z2 a1 [x0 y0]

A neutron source of relative strength **w**, uniformly distributed over the surface of a right circular cone, with its axis parallel to the z axis. The vertex of the cone is at (**x0, y0, z0**). The surface of the cone is at an angle **a1** (degrees) from the z axis, and the cone extended along the z axis between **z1** and **z2**.

Restrictions: Do not use for reactivity problems. **x0, y0** are optional input that default to 0,0. The angle **a1** MUST be greater than 0 and less than 90 degrees. Both **z1** and **z2** MUST be on the same side of **z0**. The neutrons will be launched inward, in a cosine distribution relative to the normal to the surface. The neutron source can be defined by any number **s11cone, s11cyl, s11dsk** and **s11sph** input lines.

s11coneg nz w z0 z1 z2 a1 [x0 y0]

A photon source of relative strength **w**, uniformly distributed over the surface of a right circular cone, with its axis parallel to the z axis. The vertex of the cone is at (**x0, y0, z0**). The surface of the cone is at an angle **a1** (degrees) from the z axis, and the cone extended along the z axis between **z1** and **z2**.

Restrictions: **x0, y0** are optional input that default to 0,0. The angle **a1** MUST be greater than 0 and less than 90 degrees. Both **z1** and **z2** MUST be on the same side of **z0**. The photons will be launched inward, in a cosine distribution relative to the normal to the surface. The photon source can be defined by any number of **s11coneg, s11cylg, s11dskg** and **s11sphg** input lines.

s11cyl nz w r [z0 zlen x0 y0]

A neutron source of relative strength **w**, uniformly distributed over the surface of a right circular cylinder of radius **r**, with its axis parallel to the z axis. The base of the cylinder is at (**x0, y0, z0**) and its length along the positive z axis is **zlen**.

Restrictions: Do not use for reactivity problems. **z0, x0, y0** are optional input that default to 0,0, 0. **zlen** is option input that defaults to 0, in which case the source is a circle. The neutrons will be launched inward, in a cosine distribution relative to the normal to the

surface. The neutron source can be defined by any number of **s11cone, s11cyl, s11dsk** and **s11sph** input lines.

s11cylg nz w r [z0 zlen x0 y0]

A photon source of relative strength **w**, uniformly distributed over the surface of a right circular cylinder of radius **r**, with its axis parallel to the z axis. The base of the cylinder is at (**x0, y0, z0**) and its length along the positive z axis is **zlen**.

Restrictions: **z0, x0, y0** are optional input that default to 0,0, 0. **zlen** is option input that defaults to 0, in which case the source is a circle. The photons will be launched inward, in a cosine distribution relative to the normal to the surface. The photon source can be defined by any number of **s11coneg, s11cylg, s11dskg** and **s11sphg** input lines.

s11dsk nz w ri ro [z0 d x0 y0]

A neutron source of relative strength **w**, uniformly distributed over the surface of an annulus of a circular disk of inner radius **ri** and outer radius **ro**, centered at (**x0, y0, z0**), and perpendicular to the z axis at **z0**.

Restrictions: Do not use for reactivity problems. **z0, x0, y0** are optional input that default to 0, 0, 0. **d** is option input that defaults to +1.0. The neutrons will be launched from the disk in a cosine distribution relative to the normal to the surface, in the +z direction if **d = -1.0**, or in the -z direction if **d = +1.0** (note, this is not a misprint). The neutron source can be defined by any number of **s11cone, s11cyl, s11dsk** and **s11sph** input lines.

s11dskg nz w ri ro [z0 d x0 y0]

A photon source of relative strength **w**, uniformly distributed over the surface of an annulus of a circular disk of inner radius **ri** and outer radius **ro**, centered at (**x0, y0, z0**), and perpendicular to the z axis at **z0**.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0. **d** is option input that defaults to +1.0. The photons will be launched from the disk in a cosine distribution relative to the normal to the surface, in the +z direction if **d = -1.0**, or in the -z direction if **d = +1.0** (note, this is not a misprint). The photon source can be defined by any number of **s11coneg, s11cylg, s11dskg** and **s11sphg** input lines.

s11sph nz w r [z0 x0 y0 a1 a2]

A neutron source of relative strength **w**, uniformly distributed over a segment of the surface of a sphere of radius **r**, centered at (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the positive z axis.

Restrictions: Do not use for reactivity problems. **z0, x0, y0** are optional input that default to 0, 0, 0. **a1** and **a2** are option input that default to 180, 0 degrees. The neutrons will be

launched inward, in a cosine distribution relative to the normal to the surface. The neutron source can be defined by any number of **s11cone, s11cyl, s11dsk** and **s11sph** input lines.

s11sphg nz w r [z0 x0 y0 a1 a2]

A photon source of relative strength **w**, uniformly distributed over a segment of the surface of a sphere of radius **r**, centered at (**x0, y0, z0**). The segment extends between the angles **a1** and **a2** (degrees) from the positive z axis.

Restrictions: **z0, x0, y0** are optional input that default to 0, 0, 0. **a1** and **a2** are option input that default to 180, 0 degrees. The photons will be launched inward, in a cosine distribution relative to the normal to the surface. The photon source can be defined by any number of **s11coneg, s11cylg, s11dskg** and **s11sphg** input lines.

source12 filename

A neutron source read from a file, or series of files, written by a previous problem using tally type 12.

Restrictions: Do not use for reactivity problems. **filename** is up to four characters that defines a series of files of the form NABCD000, NABCD001, etc., where N indicates neutron and ABCD are the four character **filename**. This **filename** is identical to that defined by the user with **sentl 51** in the previous problem that wrote the files. If **sentl 51** was not used in the previous problem, the default **filename** will be "cord". **Warning** - If both **source12** and **s12g** are used in a problem **sentl 18** MUST be set to 1.0, to define the relative neutron and photon strengths; otherwise the photon source will be ignored. **Warning** - If either **source12** or **s12g** files are being used, and tally type 12 is used in the problem to write files, the default filenames cannot be used for both input and output tally files; if they are used the output tally files will overwrite the input files. See **sentl 51**.

s12g filename

A photon source read from a file, or series of files, written by a previous problem using tally type 12.

Restrictions: **filename** is up to four characters that defines a series of files of the form GABCD000, GABCD001, etc., where G indicates gamma (i.e., photons) and ABCD are the four character **filename**. This **filename** is identical to that defined by the user with **sentl 51** in the previous problem that wrote the files. If **sentl 51** was not used in the previous problem, the default **filename** will be "cord". **Warning** - If both **source12** and **s12g** are used in a problem **sentl 18** MUST be set to 1.0, to define the relative neutron and photon strengths; otherwise the photon source will be ignored. **Warning** - If either **source12** or **s12g** files are being used, and tally type 12 is used in the problem to write files, the default filenames cannot be used for both input and output tally files; if they are used the output tally files will overwrite the input files. See **sentl 51**.

s12iso

This option is identical to **source12**, except that the direction cosines will be randomly sampled from an isotropic distribution.

source13 nz r [z0 x0 y0 a1 a2]

A neutron source uniformly distributed over a segment of a spherical surface of radius **r**, centered at (**x0, y0, z0**). The segment extends between **a1** and **a2** (degrees) from the positive z axis.

Restrictions: Do not use in reactivity calculations. **x0, y0, z0** are optional input that default to 0, 0, 0. **a1** and **a2** are optional input that default to 180, 0 degrees (which corresponds to a complete sphere). The reference axis for the angular distribution of neutrons is the outward normal to the spherical surface. This source can be used with multiple angle-energy spectra defined by **maec** and **maee** or **maeeh** input. Each spectrum will be generated between two cosine limits, relative to the reference axis.

s13g nz r [z0 x0 y0 a1 a2]

A photon source uniformly distributed over a segment of a spherical surface of radius **r**, centered at (**x0, y0, z0**). The segment extends between **a1** and **a2** (degrees) from the positive z axis.

Restrictions: **x0, y0, z0** are optional input that default to 0, 0, 0. **a1** and **a2** are optional input that default to 180, 0 degrees (which corresponds to a complete sphere). The reference axis for the angular distribution of photons is the outward normal to the spherical surface. This source can be used with multiple angle-energy spectra defined by **maecg** and **maeeg** or **maeehg** input. Each spectrum will be generated between two cosine limits, relative to the reference axis.

source15 nz ri ro z0 zlen a0 da [x0 y0]
s15 nz ri ro z0 zlen a0 da [x0 y0]

A neutron source uniformly distributed in a sector of a right circular cylindrical shell parallel to the z axis, with inner radius **ri** and outer radius **ro**. The base of the cylinder is at (**x0, y0, z0**), and it is of length **zlen** in the positive z direction. The sector is defined starting at an angle **a0** (degrees), in the x-y plane measured from the positive y axis toward the positive x axis. The width of the sector is defined by **da** (degrees), which may be positive or negative.

Restrictions: **a0** and **da** default to 0, so that they MUST be specified by input. **x0** and **y0** are optional input that default to 0, 0. If **zlen** = 0, the source is on a sector of an annular disk. To align the cylinder with the x or y axis use **sentl 30**. To specify a source in a

complete cylindrical shell use **source2**. The reference axis for launching neutrons is the z axis; this can be changed using **sentl 32**.

s15g nz ri ro z0 zlen a0 da [x0 y0]

A photon source uniformly distributed in a sector of a right circular cylindrical shell parallel to the z axis, with inner radius **ri** and outer radius **ro**. The base of the cylinder is at (**x0, y0, z0**), and it is of length **zlen** in the positive z direction. The sector is defined starting at an angle **a0** (degrees), in the x-y plane measured from the positive y axis toward the positive x axis. The width of the sector is defined by **da** (degrees), which may be positive or negative.

Restrictions: **a0** and **da** default to 0, so that they MUST be specified by input. **x0** and **y0** are optional input that default to 0, 0. If **zlen** = 0, the source is on a sector of an annular disk. To align the cylinder with the x or y axis use **sentl 43**. To specify a source in a complete cylindrical shell use **s2g**. The reference axis for launching neutrons is the z axis; this can be changed using **sentl 45**.

source16 nz w r [z0 a1 da]

A neutron source consisting of up to 120 circular arcs, with each arc described by its own **source16** input. The source has a relative strength **w** and is on an arc of a circle of radius **r**, centered at the point (x, y, z) = (0, 0, **z0**), in a plane perpendicular to the z axis. The arc is defined by the starting angle **a1** (degrees), measured in the x-y plane from the positive y axis toward the positive x axis. The width of the arc is defined by **da**, which may be positive or negative.

Restrictions: **z0** is optional input that defaults to 0, **a1** and **da** are optional input that default to 0, 360 degrees. To align the source with the x or y axis use **sentl 30**. The reference axis for launching neutrons is the z axis; this can be changed using **sentl 32**.

s16g nz w r [z0 a1 da]

A photon source consisting of up to 120 circular arcs, with each arc described by its own **s16g** input. The source has a relative strength **w** and is on an arc of a circle of radius **r**, centered at the point (x, y, z) = (0, 0, **z0**), in a plane perpendicular to the z axis. The arc is defined by the starting angle **a1** (degrees), measured in the x-y plane from the positive y axis toward the positive x axis. The width of the arc is defined by **da**, which may be positive or negative.

Restrictions: **z0** is optional input that defaults to 0, **a1** and **da** are optional input that default to 0, 360 degrees. To align the source with the x or y axis use **sentl 30**. The reference axis for launching photons is the z axis; this can be changed using **sentl 32**.

source17 nz r z0 x0 y0 kd

A neutron source uniformly distributed on the surface of a sphere of radius **r**, centered at (**x0, y0, z0**). The reference axis for launching neutrons will be the normal to the surface. The distribution of launch directions about the reference axis is defined by **kd**, as follows,

kd
1 - isotropic outward
2 - isotropic inward
3 - cosine distribution outward
4 - cosine distribution inward

Restrictions: **kd** MUST be specified, therefore all other parameters that precede it in the input order MUST be specified.

s17g nz r z0 x0 y0 kd

A photon source uniformly distributed on the surface of a sphere of radius **r**, centered at (**x0, y0, z0**). The reference axis for launching photons will be the normal to the surface. The distribution of launch directions about the reference axis is defined by **kd**, as follows,

kd
1 - isotropic outward
2 - isotropic inward
3 - cosine distribution outward
4 - cosine distribution outward

Restrictions: **kd** MUST be specified, therefore all other parameters that precede it in the input order MUST be specified.

New Sources

These sources can be used to sample sources from irregularly shaped zones. Unlike the other sources, these sources reject a sample if it is not inside a zone number in the range nz1 through nz2. These three new sources are for a sphere, cylinder, or rectangular box. For sampling select whichever of these shapes corresponds "best" to the shape of your actual zone numbers nz1 through nz2.

RESTRICTIONS

1) nz1 MUST be less than or equal to nz2.
2) If none of 10,000 consecutive samples from the defined volume is within zone numbers nz1 through nz2, it is assumed you made a mistake and the code will terminate. This prevents the code from going into an infinite loop of sampling and rejecting forever.

source19 nz1 thru nz2 ri ro [x0 y0 z0]
s19 nz1 thru nz2 ri ro [x0 y0 z0]
s19g nz1 thru nz2 ri ro [x0 y0 z0]

A spherical shell source of inner radius ri, and outer radius ro, centered at x0, y0, z0. Use **source19** or **s19** for neutrons, and **s19g** for photons.

nz1 - lowest zone number to sample from
nz2 - highest zone number to sample from
ri - inner radius of sphere
ro - outer radius of sphere
x0, y0, z0 - center of the sphere (optional, defaults to 0, 0, 0)

source20 nz1 thru nz2 z1 z2 ri r0 [x0 y0]
s20 nz1 thru nz2 z1 z2 ri r0 [x0 y0]
s20g nz1 thru nz2 z1 z2 ri r0 [x0 y0]

A cylindrical shell source, aligned with the z axis, extending along the z axis from z1 to z2, of inner radius ri, and outer radius ro, centered at x0, y0. Use source20 or s20 for neutrons, and s20g for photons.

nz1 - lowest zone number to sample from
nz2 - highest zone number to sample from
z1 - lower z limit of cylinder
z2 - upper z limit of cylinder
ri - inner radius of cylinder
ro - outer radius of cylinder
x0, y0 - center of the sphere (optional, defaults to 0, 0)

Note, for a cylinder aligned with an axis other than the z axis, use sentl 30 (neutrons) or sentl 43 (photons) to rotate the coordinates.

source21 nz1 thru nz2 x1 x2 y1 y2 z1 z2
s21 nz1 thru nz2 x1 x2 y1 y2 z1 z2
s21g nz1 thru nz2 x1 x2 y1 y2 z1 z2

A rectangular box in (x,y,z), extending in x from x1 to x2, in y from y1 to y2, and in z from z1 to z2. Use source21 or s21 for neutrons, and s21g for photons.

nz1 - lowest zone number to sample from
nz2 - highest zone number to sample from
x1 - lower x limit of box
x2 - upper x limit of box
y1 - lower y limit of box
y2 - upper y limit of box
z1 - lower z limit of box
z2 - upper z limit of box

There are no examples of these sources included here.

Sentinels

TART uses sentinels to define many program options. All of the sentinels are listed together here. In addition, individual sentinels have been referenced in the sections to which they apply.

Sentinels are defined by using the TART keyword **sentl** followed by a pair of parameters: first a sentinel number and then a sentinel value. In the following each sentinel will be defined in numeral order. The number in parenthesis following the definition is the default value, i.e., if you do not specify this sentinel option by input, by default this is its value. It is important for users to be aware of these default values, i.e., if you do not specify the following options by input, then by default these values are still being used in your calculation, so you had better be aware of what you are using.

The output listings of TART and TARTCHEK now include a list of all sentinels and their values for the current problem being run. Here's an example list.

```
================================================================================
 Summary of ALL Sentinels
================================================================================
  1) Type of particles (source/track)........            0
  2) # of batches to run.....................           20
  3) # of particles per batch................        50000
  4) Neutron source energy...................  0.00000D+00
  5) Neutron tally type......................            3
  6) Neutron source cosine interval..........  2.00000D+00
  7) Neutron source cosine start.............  -1.00000D+00
  8) Neutron minimum energy..................  1.00000D-11
  9) Photon minimum energy...................  -1.00000D-04
 10) Geometric uncertainty...................  1.00000D-06
 11) SPECIAL I/O filename.................... NOT ALLOWED
 12) Initial random number seed..............            0
 13) Neutron edit minimum energy.............  1.00000D-11
 14) Photon  edit minimum energy.............  1.00000D-04
 15) Photon  edit maximum energy.............  1.00000D+03
 16) Neutron edit maximum energy.............  2.00000D+01
 17) Photon  source energy...................  0.00000D+00
 18) Photons/Neutron (only if both sources)..  0.00000D+00
 19) Photon standard or all group output.....            0
 20) Multiband option........................            1
 21) Batches per long edit...................           20
 22) Mean free path edit option..............            0
 23) Reaction edit option....................            0
 24) Next problem option.....................            0
 25) Fluorescence option.....................            0
 26) Isotope order option....................            1
 27) Neutron angular dist. cosine intervals.. NOT ALLOWED
 28) Neutron energy difference edit..........            1
 29) Photon  energy difference edit..........            0
 30) Neutron source spatial axis rotation....            0
 31) Neutron source initial time.............  0.00000D+00
 32) Neutron source direction rotation.......            0
 33) Photon  tally type......................            3
 34) Neutron fission source temperature......  1.29000D+00
 35) Additional output edits.................            0
 36) Angle/energy edit.......................            0
 37) Neutron cross section edit..............            0
 38) Fission nu-bar on (0)/off (1)...........            0
 39) Thermal scattering option...............            1
 40)..................................... NOT USED
 41) Photon  source cosine interval..........  2.00000D+00
 42) Photon  source cosine start.............  -1.00000D+00
 43) Photon  source spatial axis rotation....            0
 44) Photon  source initial time.............  0.00000D+00
 45) Photon  source direction rotation.......            0
 46) Neutron standard or all group output....            0
 47) Neutron multiband edit..................            0
 48) Size of tally type 11/12 files..........           55
 49) Tally by group or only totals...........            0
 50) (n,2n) tabular kinematics...............            0
 51) Tally type 12 basic filename............ cord
 52) Relativistic kinematics.................            0
 53) Uncertainty for volume calculation......  1.00000D-08
 54) Neutrons per Fission (Nu) Method........            0
 55) Spontaneous Fission Source..............            0
 56) Tally 11/12 Only Absorption.............            0
 57) Total or Prompt nu-bar..................            0
```

Highly Recommended Options

For compatibility with earlier versions of TART for many years by default the following options are turned off, unless the user specified by input that they be turned on. Starting with TART2002 the code uses what we today consider to be the BEST available options. This effects the following sentinels

sentl 20

For neutron problems resonance self-shielding is by default turned on. This can make problems run 20 to 30 % longer, but without accounting for self-shielding the results can be completely unreliable.

sentl 25

For photon problems fluorescence is by default turned on. If no photons get down to low energies, this will have no effect on running time. However, if they do, this option is REQUIRED to obtain reliable answers.

sentl 39

For neutron problems thermal scattering is by default turned on. If no neutrons get down to thermal energies, this will have no effect on running time. However, if they do, this option is REQUIRED to obtain reliable answers.

sentl 8, 9, 13, 14, 15, 16

DO NOT USE ANY of these. These limit the energy range over which neutrons and photons are tracked and tallied. The current default option is to track neutrons and photons over the entire energy range in which cross section data is available; this is automatically determined by TART reading the binary data files.

WARNING – if you do use ANY OF THESE you can bias your results and make them invalid.

With that said, these options are still available to users because they can be used to good advantage in some special problems. For example, you may be only interested in fast neutrons, in which case you can make your problem run significantly faster by not tracking and tallying neutrons below your lowest energy of interest. Or you may want to see the effect of thermal scattering (sentl 39) or self-shielding (sentl 20) on your results. But CAVEAT EMPTOR – be warned to use these options with care.

sentl 1 (0)

Type of particles to track. This defines both the types of sources (neutrons and/or photons) and which particles to track (should neutron induced photons be tracked?).

0 - neutrons and neutron induced photons
1 - only neutrons
2 - only photons
3 - neutrons, neutron induced photons and a photon source

Restrictions: for criticality (reactivity) problems only neutrons are tracked; any **sentl 1** input is ignored. If neutron induced photons are not tracked their energy is deposited locally at the point where they are created.

sentl 2 (20)

The number of batches (samples) of particles to follow. After each batch of particles the code will output results. The total number of histories for any given run is controlled by **sentl 2 and 3**: the number of batches of particles and the number of particles per batch.

For source problems the code will always run the indicated number of batches. For criticality problems the code will run no more than the indicated number of batches; it may run less if the statistics of the problem indicate that the answer has settled to a reliable answer.

For criticality problems it is suggested that you use a much larger number of batches (200), to insure that the calculation really converges to a reliable answer. If the code runs all of the indicated number of batches before convergence it will print a warning message. The user should consider this to be a warning that the results may not be reliable. In this case it is suggested that you re-run the calculation using many more batches (2000).

sentl 3 (5000)

The number of particles per batch (sample). The total number of histories for any given run is controlled by **sentl 2 and 3**: the number of batches of particles times the number of particles per batch.

It is recommended that you not use a small number of particles per batch.

For source problems the final statistics will be based on the total number of histories run = the number of batches times the number of particles per batch. In this case each history will be run completely independently. At the end of each batch the code will print complete results, so that the output file will be very large if you run a large number of batches. A good compromise is to first decide the total number of particles that you want to run and then run about 10 batches, each batch being about 1/10-th of the total number of histories you want to run.

For criticality problems the number of completely independent histories run is defined by the number of histories per batch. The same histories are tracked generation after

generation until the distribution settles into an equilibrium distribution (in space, energy and direction). You will find that the time to convergence is related to the number of batches run times the number of neutrons per batch. In this case you should not use a small number of histories per batch. If you do the statistics for any given batch will be extremely poor and the code will have to run many batches in order to converge. The result will be that you will not save much time, compared to running larger batches of particles, i.e., using larger batches, convergence will occur after running a smaller number of batches.

sentl 4 (0.0)

Neutron source energy. For photons use **sentl 17**.

Restrictions: The default value of 0.0 indicates a neutron induced fission neutron spectrum. This is used for monoenergetic sources. If a spectrum is input this option is ignored. The neutron source energy must be within the range of TART multigroups = 1.0e-11 to 20 MeV for 616 groups.

sentl 5 (3)

Neutron tally type. For photons use **sentl 33**.

Restrictions: The default value (3) is neutrons entering each zone. See also, **ltype, probm, probmv, units, sentl 35, sentl 36.**

sentl 6 (2.0)

The cosine interval of the neutron source angular distribution. For photons use **sentl 41**.

The range of the angular distribution of the neutron source is defined by **sentl 6 and 7**. The angular distribution relative to the z axis, extended from **sentl 7** to **sentl 7** plus **sentl 6**. With the default values for **sentl 6** and **7 of** 2.0 and -1.0 the distribution extends from a cosine of -1.0 to a cosine of +1.0 = an isotropic angular distribution. For a monodirectional source directed straight up the z axis, use **sentl 7 1.0** and **sentl 6 0.0**. See also, **anglsrce, sentl 32.**

Restrictions: The entire cosine range from **sentl 7** to **sentl 7** plus **sentl 6**, cannot extend outside the legal cosine range -1.0 to +1.0.

sentl 7 (-1.0)

The start of the range of the neutron source angular distribution. For photons use **sentl 42**.

The range of the angular distribution of the neutron source is defined by **sentl 6 and 7**. The angular distribution relative to the z axis, extended from **sentl 7** to **sentl 7** plus **sentl**

6. With the default values for **sentl 6** and **7 of** 2.0 and -1.0 the distribution extends from a cosine of -1.0 to a cosine of +1.0 = an isotropic angular distribution. For a monodirectional source directed straight up the z axis, use **sentl 7 1.0** and **sentl 6 0.0**. See also, **anglsrce, sentl 32.**

Restrictions: The entire cosine range from **sentl 7** to **sentl 7** plus **sentl 6**, cannot extend outside the legal cosine range -1.0 to +1.0.

sentl 8 (1.0d-11 MeV)

Minimum neutron energy. For photons use **sentl 9**. DO NOT USE THIS!

For a positive minimum neutron energy any neutron with an energy less than this will have its energy set equal to the minimum. For a negative minimum neutron energy any neutron with an energy less than this will be discarded. When the default option is used neutrons are returned to the minimum energy.

When using thermal scattering, **sentl 39**, this option should be set to the minimum energy of the TART multigroup data = 1.0e-11 for 616 groups. Failure to do so will prevent neutrons from fully thermalizing, e.g., for a room temperature problem and the default value of 2.53e-08, the neutrons cannot scatter to energies below the average Maxwellian energy of the target nuclei, so that they cannot fully thermalize.

Restrictions: The neutron source energy must be within the range of the TART multigroups = 1.0e-11 to 20 MeV for 616 groups.

WARNING - DO NOT use this option, unless you really want to limit the minimum energy of neutrons. TART will now use the minimum neutron energy of the data read from the data files - which is currently 1.0d-11 MeV.

sentl 9 (-1.0e-04)

Minimum photon energy. For neutrons see **sentl 8**. DO NOT USE THIS!

For a positive minimum photon energy any photon with an energy less than this will have its energy set equal to the minimum. For a negative minimum photon energy any photon with an energy less than this will be discarded. When the default option is used, photons are discarded.

Restrictions: The photon source energy must be within the range of the TART photon data = 1.0e-04 to 30.0 MeV.

WARNING - DO NOT use this option, unless you really want to limit the minimum energy of photons. TART will now use the minimum photon energy of the data read from the data files.

sentl 10 (1.0e-06)

Geometric uncertainty; referred to as **FUDGE.** When advancing to a boundary this amount will be added to the distance to the boundary to move the particle off the boundary and into a new zone.

Restrictions: This option must be positive and should be small compared to the dimensions, cm, of any zone in a problem. If this option is larger than the dimensions of a zone in a problem advancing particles by this amount can cause them to move completely across the zone and never "see" the zone. For most applications the default value is adequate; only if you are dealing with either extremely small or extremely large systems should you define a different value for this option.

sentl 11 (blank)

The filename of a file in which special input and output data is stored. See **"sentl 11 Data File Contents"**.

Restrictions: The filename cannot exceed 8 characters; if it does, it will be truncated to 8 characters. The contents of the TART files differs from the contents of the standard TARTNP files. Contact the author for a copy of the FORTRAN routines required to read the TART files.

sentl 12 (0)

Random number sequence index. TART now has 2,510 sequences, one trillion (10^{12}) samples apart. Input 0 (the default) to 2509 will use the selected sequence. Any other input is a fatal ERROR.

If you run the same problem more than once using the same random number sequence, you will always get exactly the same answer. **Warning** - you should never try to use this option (0) to run the same problem more than once to improve statistics. If you do, you will merely be running exactly the same histories over again and get exactly the same - exactly correlated - results, which will not improve the accuracy of your results.

You can use this option in conjunction with the MULTIPRO and TARTSUM utility codes to improve statistics, by making a series of run each with a difference random number sequence.

sentl 13 (1.0d-11 MeV)

Minimum energy for editing the coordinates of neutrons entering a zone when using tally type 11 or 12. For photons see **sentl 14**. DO NOT USE THIS!

Restrictions: The energy must be within the range of the TART multigroups = 1.0e-11 to 20 MeV for 616 groups.

WARNING - DO NOT use this option, unless you really want to limit the minimum tally energy of neutrons. TART will now use the minimum neutron energy of the data read from the data files - which is 1.0e-11 MeV for 616 groups.

sentl 14 (1.0e-04)

Minimum energy for editing the coordinates of photons entering a zone when using tally type 11 or 12. For neutrons see **sentl 13**. DO NOT USE THIS!

Restrictions: The energy must be within the range of the TART photon data = 1.0e-04 to 30.0 MeV.

WARNING - DO NOT use this option, unless you really want to limit the minimum tally energy of photons. TART will now use the minimum photon energy of the data read from the data files.

sentl 15 (1000.0)

Maximum energy for editing the coordinates of photons entering a zone when using tally type 11 or 12. For neutrons see **sentl 16**. DO NOT USE THIS!

Restrictions: The energy must be within the range of the TART photon data = 1.0e-04 to 30.0 MeV. If your photon sources energies extend above 20 MeV (they can be up to 30 MeV), failure to increase this option to your maximum photon source energy will cause photons above 20 MeV not to contribute to tallies.

WARNING - DO NOT use this option, unless you really want to limit the maximum tally energy of photons. TART will now use the maximum photon energy of the data read from the data files.

sentl 16 (20.0)

Maximum energy for editing the coordinates of neutrons entering a zone when using tally type 11 or 12. For photons see **sentl 15**. DO NOT USE THIS!

Restrictions: The energy must be within the range of the TART multigroups = 1.0e-11 to 20 MeV for 616 groups.

WARNING - DO NOT use this option, unless you really want to limit the maximum tally energy of neutrons. TART will now use the maximum neutron energy of the data read from the data files.

sentl 17 (0.0)

Photon source energy. For neutrons see **sentl 4**.

Restrictions: The default value of 0.0 indicates a fission spectrum. This is used for monoenergetic sources. If a spectrum is input this option is ignored. The energy must be within the range of the TART photon data = 1.0e-04 to 30.0 MeV.

sentl 18 (0.0)

The number of source photons to follow per source neutron. This option is only used when **sentl 1 = 3** (track both neutron and independent photon sources) or when using both **source12** and **s12g**, where the neutron and photon source distributions are read from files.

Restrictions: when **sentl 1 = 3** this option defines the relative strength of the neutron and photon sources. In this case failure to define this option can cause photon sources to be ignored. For example if this option is 1.0, for each source neutron tracked, one source photon will be tracked. When using both **source12** and **s12g** this option **MUST** be set to 1.0, to insure that all source neutrons and photons, read from the files, are tracked.

sentl 19 (0)

Photon tally group sentinel. Photon transport represents cross sections in 700 energy intervals between 100 eV and 1 GeV, using 100 equally spaced log intervals per energy decade. This option can be used to select how many of these 700 intervals are used to tally results,

0 = 70 photon tally bins (every 10 energy intervals = default).
1 = 175 photon tally bins (every 4 energy intervals).
2 = 700 photon tally bins (every energy interval).

See **cphotal** and **ltypeg**. For neutrons use **sentl 46**.

sentl 20 (1)

Multiband option. TART uses multigroup cross sections that do not include the effects of resonance self-shielded; so called, unshielded cross sections. Setting this option to 1 will cause the code to use the multiband method to account for the effects of resonance self-shielding.

Recommendation: **It is highly recommended that you NEVER turn this option off.**

sentl 21 (20)

Number of batches (samples) for which short edits are output between each batch (sample) for which a long edit is output. A long edit is always output for the first and last batches (sample). Use -1 to get a long edit of every batch (sample). Use 0 or 1 to get a short edit every other batch (sample).

A short edit includes only a brief summary of balance or economy results summed over batches run so far, e.g., energy leaked, etc. A long edit include this brief summary plus all of the output requested by user input.

sentl 22 (0)

Mean free path edit sentinel. If 1, the code will not edit the mean free paths, cross section probabilities and energy deposition per collision.

sentl 23 (0)

Reaction edit sentinel. If 1, edits requested using the **reacted** option will not include reactions for each energy group; only the total for each reaction for each zone with be edited.

sentl 24 (0)

Next problem sentinel. If 1, input data for another problem follows the **end** input line for the current problem. (no longer used for TART 2002 and later – the code will continue reading input to the end of the input file).

If you wish to run more than one problem this option **MUST** be set. Otherwise the code will terminate at the end of the current problem.

sentl 25 (1)

Fluorescence sentinel. If 1, photons produced by fluorescence will be tracked. Otherwise their energy will be deposited locally at the spatial point where they are created.

Recommendation: **It is highly recommended that you NEVER turn this option off.**

sentl 26 (1)

Isotope reordering sentinel. If 2, for each material in the problem isotopes will be reordered in descending order by atomic abundance, to improve sampling efficiency.

Recommendation; It is recommended that when you define the materials for use in a problem, you define the order of isotopes in each material in descending order of importance. **sentl 26** can order isotopes based on atomic abundance, which is not necessarily the correct order by importance, since the cross sections for one isotope may be considerable larger or smaller than those of another isotope. You are in a much better position to decide what is or is not important in your problems, than this code is.

sentl 27 (32)

The number of neutron anisotropic scattering angular distribution intervals. The option values are 16, 8 or 4.

Recommendation: This option is no longer allowed. The tabulated neutron anisotropic scattering angular distributions in the neutron interaction data file (TARTND) have been calculated to define equally probable scattering cosine intervals that are sampled whenever a scattering event occurs. By changing to a smaller number of intervals all you will do is end up with a worse approximation to the original evaluated data.

sentl 28 (0)

Neutron energy-difference edit sentinel. For photons use **sentl 29**. If 1, the edits of each tally for each zone by energy and time will be repeated with each tally divided by the energy width of each tally group. If 2, the tallies will be multiplied by an input constant for each tally group, as specified on the **edif** input option. Note, if the **edif** input option is used, **sentl 28** will automatically be set to 2. This output will allow you to obtain energy dependent responses from the user supplied energy dependent table per MeV, as opposed to simply integrating over each energy tally bin.

Restriction: If this option is 2, you MUST use **edif** input. If using the **edif** input option, you need not set this option to 2, but you **MUST** not set it equally to any other value.

sentl 29 (0)

Photon energy-difference edit sentinel. For neutrons use **sentl 28**. If 1, the edits of each tally for each zone by energy and time will be repeated with each tally divided by the energy width of each tally group. If 2, the tallies will be multiplied by an input constant for each tally group, as specified on the **edifg** input option. Note, if the **edifg** input option is used, **sentl 29** will automatically be set to 2.

Restriction: If this option is 2, you MUST use **edifg** input. If using the **edifg** input option, you need not set this option to 2, but you **MUST** not set it equal to any other value.

sentl 30 (0)

Neutron source spatial coordinate transformation. For photons use **sentl 43**. When this option is used ALL source neutrons will have their initial spatial coordinates (X, Y, Z) switched are follows,

0 - (X, Y, Z) = No change
1 - (X, Z, Y) = Exchange Y and Z
2 - (Z, Y, X) = Exchange X and Z
3 - (X, -Z, Y) = Reverse the sign of Z, exchange Y and Z
4 - (-Z, Y, X) = Reverse the sign of Z, exchange X and Z
5 - (X, Y, -Z) = Reverse the sign of Z

Restrictions: Cannot be used for criticality problems, where the **critcalc** input option is used. **source3, source4** or **source9** will not be effected by this option if an **axis** input option is used.

sentl 31 (0.0)

Initial time coordinate of source neutrons.

sentl 32 (0)

Neutron source direction transformation. For photons use **sentl 45**. When this option is used ALL source neutrons will have their initial direction cosines (alpha, beta, gamma), switched are follows,

0 - (alpha, beta, gamma) = No change
1 - (beta,gamma,alpha) = Rotate left
2 - (gamma,alpha,beta) = Rotate right
3 - (beta,-gamma,alpha) = Reverse sign of gamma, rotate left
4 - (-gamma,alpha,beta) = Reverse sign of gamma, rotate right
5 - (alpha,beta,-gamma) = Reverse the sign of gamma
6 - (-gamma,beta,alpha) = Rotation about y axis; use with **sentl 30 4**

Restrictions: **source3, source4** or **source9** will not be effected by this option if an **axis** input option is used.

sentl 33 (3)

Photon tally type. For neutrons use **sentl 5**.

Restrictions: The default value (3) is photons entering each zone. See also, **ltypeg, probm, probmv, units, sentl 35, sentl 36.**

sentl 34 (1.29 MeV)

Temperature of the source neutron induced neutron fission spectrum.

Restrictions: The temperature MUST be positive. Only used for source neutron induced neutron fission spectrum, i.e., **sentl 4 = 0.0**. When this option is used the source neutron fission spectrum built into TART will be rescaled to have an average energy corresponding to the input value.

Recommendations: **Warning** - unless you are very familiar with fission spectra and understand the consequences of your action, it is highly recommended that you not use this option. There is no equivalent for source neutron induced photon fission spectrum.

sentl 35 (0)

Sentinel for additional edits.

1
An additional edit will be made for each problem multiplier specified on a **probm** or **probmv** input option, with the output multiplied by the problem multiplier.

2
The same as using option 1, but the output will be divided by the volume of each zone.

3
An additional edit will be made with the output multiplied by the zone multipliers specified on **units (zonemult)** input lines. An additional edit will be made for each problem multiplier, with the output multiplied by the problem multiplier and multiplier for each zone.

4
The same as using option 3, but the output will be divided by the volume of each zone.

5
The same as using option 3, but the output will be multiplied by the energy group multipliers specified on **edif** input lines, or divided by the energy width of each tally group, if **sentl 28 = 1**.

6
The same as using option 2, but the output will be divided by the mass of each zone, rather than the volume.

7
The same as using option 3, but the output will be divided by the mass of each zone, rather than the volume.

Restrictions: Depending on the option selected you may have to also input **probm** or **probmv, units (zonemult)** or **edif**. TART can only correctly calculate volumes, and therefore masses, for zones that are symmetric about the Z axis; in all other cases it arbitrary defines the volume of zones to be 1.0. Therefore in all other cases using this option to request output involving the volume and/or mass of zones will produce misleading results. Whenever you use this option please check the calculated volume for each zone, before relying on the accuracy of the results.

sentl 36 (0)

Sentinel for additional angular distribution edits, that are tallied by angle and energy for zones with tally types 7-10. The options are identical to those of **sentl 35.**

Restrictions: Same as for **sentl 35**.

sentl 37 (0)

Neutron cross section edit sentinel. If 1, output will include an edit of the neutron cross sections for each isotope in the problem by reaction type and energy.

sentl 38 (0)

Fission nu-bar sentinel. If 1, ALL fission nu-bar will be set to zero, so that no neutrons will be produced by fission.

If you have a final distribution of neutrons that already includes the effect of all fission neutrons produced you can use this option to transport them further and determine the effect that this distribution has.

sentl 39 (1)

Thermal scattering sentinel. If 1, thermal scattering will be used, otherwise all target nuclei will be assumed to be stationary.

The temperature of each zone should be defined by **emin** input; if not defined by input the default 2.53e-08 MeV (room temperature) will be used.

Restrictions: WARNING – Please no longer use sentl 8 and 13; these energies will now be automatically set to the minimum neutron energy read from the neutron data file.

Recommendation: **It is highly recommended that you NEVER turn this option off.**

sentl 40

Not used.

sentl 41 (2.0)

The cosine interval of the photon source angular distribution. For neutrons use **sentl 6**.

The range of the angular distribution of the photon source is defined by **sentl 41** and **42**. The angular distribution relative to the z axis, extended from **sentl 42** to **sentl 42** plus **sentl 41**. With the default values for **sentl 41** and **42** of 2.0 and -1.0 the distribution extends from a cosine of -1.0 to a cosine of +1.0 = an isotropic angular distribution. For a monodirectional source directed straight up the z axis, use **sentl 42 1.0** and **sentl 41 0.0**. See also, **anglsrg, sentl 45.**

Restrictions: The entire cosine range from **sentl 42** to **sentl 42** plus **sentl 41**, cannot extend outside the legal cosine range -1.0 to +1.0.

sentl 42 (-1.0)

The start of the range of the photon source angular distribution. For neutrons use **sentl 7**.

The range of the angular distribution of the photon source is defined by **sentl 41** and **42**. The angular distribution relative to the z axis, extended from **sentl 42** to **sentl 42** plus **sentl 41**. With the default values for **sentl 41** and **42** of 2.0 and -1.0 the distribution extends from a cosine of -1.0 to a cosine of +1.0 = an isotropic angular distribution. For a monodirectional source directed straight up the z axis, use **sentl 42 1.0** and **sentl 41 0.0**. See also, **anglsrg, sentl 45.**

Restrictions: The entire cosine range from **sentl 42** to **sentl 42** plus **sentl 41**, cannot extend outside the legal cosine range -1.0 to +1.0.

sentl 43 (0)

Photon source spatial coordinate transformation. For neutrons use **sentl 30**. When this option is used ALL source photons will have their initial spatial coordinates (X, Y, Z) switched are follows,

0 - (X, Y, Z) = No change
1 - (X, Z, Y) = Exchange Y and Z
2 - (Z, Y, X) = Exchange X and Z
3 - (X, -Z, Y) = Reverse the sign of Z, exchange Y and Z
4 - (-Z, Y, X) = Reverse the sign of Z, exchange X and Z
5 - (X, Y, -Z) = Reverse the sign of Z

Restrictions: **s3g, s4g** or **s9g** will not be effected by this option if an **axisg** input option is used.

sentl 44 (0.0)

Initial time coordinate for source photons. For neutrons use **sentl 31**.

sentl 45 (0)

Photon source direction transformation. For neutrons use **sentl 32**. When this option is used ALL source photons will have their initial direction cosines (alpha, beta, gamma), switched are follows,

0 - (alpha, beta, gamma) = No change
1 - (beta,gamma,alpha) = Rotate left
2 - (gamma,alpha,beta) = Rotate right
3 - (beta,-gamma,alpha) = Reverse sign of gamma, rotate left
4 - (-gamma,alpha,beta) = Reverse sign of gamma, rotate right
5 - (alpha,beta,-gamma) = Reverse the sign of gamma

6 - (-gamma,beta,alpha) = Rotation about y axis; use with **sentl 43 4**

Restrictions: **s3, s4g** or **s9g** will not be effected by this option if an **axisg** input option is used.

sentl 46 (0)

TART2002 Update

Neutron tally group sentinel. Neutron transport represents cross sections in 700 energy intervals between 10^{-5} eV and 1 GeV, using 50 equally spaced log intervals per energy decade. Currently only 616 groups, up to 20 MeV are used. This option can be used to select how many of these 616 intervals are used to tally results,

0 = 80 neutron tally bins (every 10 energy intervals, plus all above 10 MeV = default).
1 = 165 neutron tally bins (every 4 energy intervals, plus all above MeV).
2 = 616 neutron tally bins (every energy interval).

See **cneutal, reactall, reacted** and **ltype**. For photons use **sentl 19**.

sentl 47 (0)

Multiband edit sentinel (see **sentl 20**). If 0, standard edits (totals by reaction). If 1, extra tallies are edited: tritons produced per zone, (n,gamma) reactions by zone and isotope, and fission reactions by zone and isotope.

sentl 48 (55)

The size of the tally type 11 and 12 coordinate file will be changed to this option times 10,000 plus 20 words. See **sentl 51**.

sentl 49 (0)

Tally type energy group deletion sentinel.

If 1, no output by energy group, but output the totals and standard deviation of energy deposits, zone tallies and reactions, for each zone and time step.

If 2, the same as 1 without zone tallies.

If 3, the same as 1 with a standard full tally for the last sample. This option can be used to minimize output for intermediate batches and still allow you to obtain complete results at the end of the run. See **notal**.

sentl 50 (0)

(n,2n) tabular kinematics sentinel. If 1, use tabular kinematics for all (n,2n) reactions except for Be^9 (ZA = 4009).

sentl 51 ("cord")

Tally type 12 coordinate filename basis; an alphanumeric string of up to 4 characters. The filenames will be NABCD000 (neutrons) or GABCD000 (photons), where ABCD = the 4 characters input for this option.

Restriction: Use different filenames for **source12** or **s12g** input files and tally type 12 output, or the input files may be overwritten.

sentl 52 (0)

Relativistic kinematics sentinel. Not currently used.

sentl 53 (1.0e-08)

Geometric uncertainty (Fudge) used in defining zone volumes.

sentl 54 (0)

The definition of the number of neutrons per fission. If this option is 0 (the default), the average number of neutrons per fission, ν(E), is used. If this option is 1, the energy distribution of the number of neutrons per fission is sampled; this later option is designed for use in reactor noise analysis calculations.

Restrictions: This option can only be used for neutron source problems; it cannot be used for criticality calculations. If you try to use this option for criticality calculations the code will abort.

sentl 55 (0)

Spontaneous fission source. If this option is 0 (the default) all source neutrons are treated as uncorrelated, as far as time and position of emission, and you may specify the neutron source energy spectrum by input. If this option is positive (see, restrictions below), the neutron source is treated as a spontaneous fission source, with the emission of neutrons correlated, as far as time and position of emission. The total source strength for any problem is still defined by the product of sentinels 2 and 3, however when this option is used, the number of neutrons per fission is used to emit, ν, successive neutrons at the same time and position; the neutron energies and directions are sampled independently from the appropriate distributions.

Example input: sentl 55 98252, will cause the neutron source to be treated as a Cf252 spontaneous fission spectrum.

Restrictions: This option can only be used for neutron source problems; it cannot be used for criticality calculations. If you try to use this option for criticality calculations the code will abort. Currently this option is limited to ZA = 94240 (Pu240), 96242 (Cm242), 96244 (Cm244), and 98252 (Cf252). When you use this option, you MUST use a neutron fission spectrum source (sentl 4 default), and the angular distribution MUST be isotropic (sentl 6 and 7 defaults).

sentl 56 (0)

Tally 11 and 12 in zone absorption. Normally when using tally type 11 or 12 for neutrons, every time a neutron enters the zone its coordinates are tallied in a binary file. When this option is 1, neutron coordinates are only tallied when a neutron is absorbed (capture or fission) in the zone. This option can be used to better simulate neutrons detector response, where the response is due to neutron absorption.

TART2002 update

This option has been extended to use tally 11 and 12 to tally,
0 = Leakage (the **number of neutrons** entering a zone)
1 = Absorption (Capture + Fission: neutrons absorption **events** in a zone)
2 = Production (the **number of neutrons** ($<\nu>$) produced in a zone)
3 = Capture (neutrons captured **events** in a zone)
4 = Production (neutron production **events** in a zone)

With these options you can tally all of the ways of producing and removing neutrons. If you do tallies for all zones you can obtain conservation relationships for the system as a whole, e.g., the sum of the number of source neutrons plus the number of neutrons produced MUST EXACTLY equal the sum of neutrons that are removed by leakage and absorption (absorption = capture + fission).

Here's a few definitions,
Absorption (1) = Capture (3) + Fission (4)
Removal = Leakage (0) + Absorption (1)
Removal = Source + Number Produced (2)
Number Produced (2) = $<\nu>$ times Production Events (4)

For any sub-critical system you can also derive,
total multiplication = [Removal]/[Source]
net multiplication = [Removal – Production (4)]/[Source]

Restrictions: During any single TART run sentl 56 can only have one unique value. Therefore if you want to tally more than one type of information you will have to make more than one TART run. The use of one value during one run does not interfere with the results of another value for a separate run, or change the random number sequence during a run. Therefore you can expect exact correspondence between the results of several runs tallying different quantities for the same problem, e.g., you can expect that the sum of all

neutron sources (initial source plus fission production) will be exactly equal to the sum of all neutron removal (leakage plus absorption).

sentl 57 (0)

Select either total nu-bar (prompt plus delayed) or prompt (ignore delayed). The default is total nu-bar. This option can be used to better simulate systems that are strongly time dependent, in which case the multiplication of the system is better characterized by the prompt nu-bar.

Restrictions: the only restrict is that you MUST use a recent TARTND nuclear data file (dated 11/22/00 or later), since only these file include both total and prompt nu-bar.

Neutron and Photon Edit and Tally (Output) types

Regardless of how fast and accurate a computer code is, its not very useful to you if you cannot obtain the output results that you want for your applications; this is the real proof of the pudding - can it solve your problem and give you the answers you want? One of the most powerful features of TART is the wide variety of tally and output options available to the user. To be able to take full advantage of the features of TART the user should be aware of what the standard TART default output includes and the availability of the following types of output options that you can specify by input.

Default Tally and Output

Generally output will be produced for every zone in a problem for each type of particle being tracked (neutrons and/or photons). The code will always output energy deposition for each zone. For all zones that contain material (all interior zones) particles will be tracked and both expected and analog deposition will be output. For all zones that do not contain material (exterior zones) the history will be terminated when the particle enters the zones and all of the particle's energy will be scored as deposition. In this case only the analog results will be output. This deposition will only be one integral number per zone, i.e., the results are not energy dependent.

If the user does not define what to score (tally) by default this is equivalent to specifying **ltype 3** (for neutrons) or **ltypeg 3** (for photons) which is particles entering each zone. Note, that this default of particles entering each zone, is the current entering the zone, not the flux. If you would prefer flux within each zone you need merely specify by input **ltype 2** (for neutrons) or **ltypeg 2** (for photons). These results are output both as a single integral number per zone as well as in energy dependent form using 50 tally groups for neutrons or photons. For details of the default 50 tally groups see tables 3 and 5 for neutrons and photons, respectively.

The user has the input option to change tally energy groups, but these cannot be arbitrarily specified. In all cases the boundaries of tally groups MUST be a subset of the 616 neutron group boundaries or 700 photon energy intervals and MUST span the energy range of all particles in a problem. To change the tally groups for neutrons use **cneutal** and for photons use **cphotal**. These options can be used to specify up to 616 (neutron) or 700 (photon) new tally groups, respectively. For neutrons you can use 616 or 175 tally groups, rather than 80, using **sentl 46**. For photons you can use 700 or 175 tally groups, rather than 70, using **sentl 19**.

Angular Distribution Tally and Output

Normally output will not include the angular distribution of particles. To obtain angular distributions use **ltype** (for neutrons) or **ltypeg** (for photons), equal to **7**, **8**, **9**, **10**. In this case you will obtain all other tallies requested PLUS this additional angular distribution output. However, in this case the neutron output is limited to 50 tally groups; you cannot

obtain output in 175 tally groups. Angular distributions output will use 19 cosine bins; see table 8 for the cosine limits of these bins. To change the cosine bins use **cnang** (for neutrons) and **cpang** (for photons). For angular distributions the number of energy tally groups will normally be decreased from 50 to 12; see table 8 for the energies of the 12 neutron and photon tally groups. For angular distribution output to change from the standard 12 groups use **angn** (for neutrons) and **angg** (for photon). When these options are used you can specify up to 24 energy tally bins, i.e., up to 25 tally limits.

Time Dependent Output

Normally if you do not specify anything by input the default is that all results are integrated over all time. As such the output from TART is NOT flux, it is fluence, which is flux integrated over all time. Similarly the energy deposition is the deposition integrated over all time. "All time" is defined as one second, 1.0e+8 shakes, which exceeds the half-life of a free neutron.

To obtain time dependent output use **centim** which applies to both neutrons and photons. When this option is used the output will be results integrated over the time intervals specified by the **centim** input.

WARNING - Once particles reach the maximum census time they are terminated. Therefore this is an effective means of limiting how long particles are followed. However, users MUST insure that the initial time of all source particles does not exceed the maximum census time of the problem. Otherwise, results may be inconsistent.

Response Functions

You can obtain additional output which is defined as the product of what you are tallying times an energy dependent response function. Use **edif** (for neutrons) and **edifg** (for photon). Results can be output in the form of either tally bin integrals (the default) or tally bin averages; for averages use **sentl 28** (for neutrons) and **sentl 29** (for photons).

This is a completely general means of modifying the output that you obtain. But the interpretation of the output up to the user and depends both on what you specify as the multiplier and what you specify as the scoring option. For example, if for neutrons you are scoring path length in each zone (**ltype 2**) and your **edif** input is a group averaged cross section the result will be the reactions (the product of cross section times fluence). You can use this option to calculate dose, a simple detector response, virtually anything that is proportional to what you are tallying. The only limit on the usefulness of this option is the imagination of the user.

The simplest use of this option is to merely change the units of your output. For example, normally the TART fluence (path length) output is in centimeters. If you would like output in any other units you can simply specify an energy independent multiplier.

Problem Multipliers

The response function capability described above is useful when you want to specify energy dependent multipliers for what you are tallying. The problem multiplier options **probm** and **probmv** can be used to define a series of energy independent multipliers to obtain results for energy deposition and fissions per unit mass (**probm**) or volume (**probmv**). Used in conjunction with **sentl 35** and **36** the results can be specified to be either integral or average values.

The simplest use of this option is to merely change the units of your output; you can obtain output in as many different sets of options as you want. For example, normally the TART energy deposition output is in MeV. If you would like output in any other units you can simply specify an energy independent multiplier.

Zone Multipliers

The problem multipliers defined above are applied uniformly to all zones. If you would like to specify a multiplier for individual zones use the option **units** or **zonemult** (these are equivalent). These zone multipliers can be used in conjunction with the problem multipliers, described above. These zone multipliers apply to everything that you tally (not just energy deposition and fissions) and produces additional output listings, i.e., it does not replace the basic output, it supplements it.

Cross Section and Reaction Output

As a default you will always obtain output of the mean free path, probability of individual reactions (i.e., cross section ratio to the total) and expected energy deposition for each material used in a problem; not each evaluation, only each material, e.g., if you use hydrogen and oxygen to define water you will only obtain output of the total cross section for water. The mean free path and energy deposition are macroscopic quantities based on the total cross section and deposition as well as the density specified for each individual material.

If you would to obtain a listing of all cross sections for all groups, reactions and isotopes used in a problem use **sentl 37**. **Warning** - for an average problem this option is not recommended, since it can result in a large amount of output.

If you would like output of the individual reactions that actually occur during the neutron Monte Carlo use **reacted** or **reactall**. **reacted** allows you to request for one zone the reaction with individual isotopes. You can use any number of **reacted** input lines to obtain output for as many isotopes in as many zones as you want. **reactall** allows you to request reactions for all isotopes in a list of zones. Anytime that you request **reacted** or **reactall** output the neutron tally groups will be changed from 80 to 616 groups and normally **reacted** and **reactall** output will be energy dependent by tally groups. If you would prefer only the total number of each reaction (integrated over tally groups) for each zone use **sentl 23**.

Additional and Non-Standard Tally and Output

You will always obtain a standard set of output results, as described above. You can modified what you tally and specify additional output results. An important point to understand is that if you request any additional output, it really is additional - it does not replace the basic output, that you will also receive.

In addition the user may specify additional types of output. The default option is in all cases 3 = the number of particles entering into each zone. This option may be changed zone by zone using **ltype** for neutrons or **ltypeg** for photons. For neutrons it may be changed globally using **sentl 5** for all zones that **ltype** is not specified, as it can using **sentl 33** for photons for all zones that **ltypeg** is not specified. This is a very powerful and versatile capability that the user should be aware of.

ltype l nz1 nz2 nz3....(3)
ltype l nza thru nz2 (3)

Neutron edit type. **ltype** can be used to define the neutron edit type zone by zone; **sentl 5** allows it to be defined globally as the same value for all zones.

Restrictions: The default is 3 (particles entering the zone); see the complete list of choices, later in this report.

WARNING – when using ltype 11 or 12 to write binary files of neutron histories, be aware that the zone(s) that you wish to tally in MUST contain material; otherwise, when a particle enters an empty zone it is immediately absorbed, without allowing ltype 11 or 12 tally.

Starting with TART99 ltype 11 or 12 can be used for empty zones (zones to which no material is assigned); when a particle enters an empty zone, tally type 11 or 12 scores are checked for before absorbing the particle. This is a convenient option to use if you want to insure that each particle can result in at most one tally.

ltypeg l nz1 nz2 nz3....(3)
ltypeg l nza thru nz2 (3)

Photon edit type. **ltypeg** can be used to define the neutron edit type zone by zone; **sentl 33** allows it to be defined globally as the same value for all zones.

Restrictions: The default is 3 (particles entering the zone); see the complete list of choices, later in this report.

WARNING – when using ltypeg 11 or 12 to write binary files of photon histories, be aware that the zone(s) that you wish to tally in MUST contain material; otherwise,

when a particle enters an empty zone it is immediately absorbed, without allowing ltypeg 11 or 12 tally.

Starting with TART99 ltypeg 11 or 12 can be used for empty zones (zones to which no material is assigned); when a particle enters an empty zone, tally type 11 or 12 scores are checked for before absorbing the particle. This is a convenient option to use if you want to insure that each particle can result in at most one tally.

notal j nz1 nz2 nz3....(0)
notal j nz1 thru nz2 (0)

notal (no tally) can be used to indicate zones that should not be edited (no output). With the default value, 0, the zone will be edited by energy group and time. If 1, the zone will not be edited.

For example, in a problem that has 400 zones (numbered 1 through 400), if you only want output for zones 1, 2, 7, 10 and 100, the following input can be used,

notal 1 3 thru 400
notal 0 7 10 100

Note, the first line indicates no tallies for all zones except zones 1 and 2. The second line then indicates tallies for zones 7, 10 and 100.

Restrictions: The default is 0 = edit all zones. Setting **sentl 49 1** will give tallies only in zones where the **notal** option is 0, but setting **sentl 49 2** overrides **notal** and no zones will be edited by tally type.

The available types of tallies include,

1 - Expected collisions (MUST be used to get reaction edits)
2 - Path length in centimeters
3 - Number of particles entering a zone (DEFAULT). This is the only possible tally type for a void zone, e.g., the exterior, non-re-entrant zone.
4 - Energy transported into a zone
5 - Flux entering a zone = number divided by cosine to normal
6 - Energy flux entering a zone
7 - Same as 3 with angular distribution
8 - Same as 5 with angular distribution
9 - Same as 4 with angular distribution
10 - Same as 6 with angular distribution
11 - Writes a disk file
12 - Writes a disk file
13 - Expected energy deposited by collision (photons only)
14 - Actual energy deposited by collision (photons only)

15 - Number of particles at census time (neutrons only)
16 - Write a disk file when entering a zone.
17 - Write a disk file after every collision in the zone.
18 - Collect particles that enter this zone, move them to a collector surface, and relaunch them, See **t1819, t18sph, t18dpl.**
19 - Collect particles that enter this zone, move them to a collector surface, and relaunch them in the +Z direction, See **t1819**.
20 - Number of Monte Carlo collisions
21 - Number of photons that deposit energy in the zone (photons only).

Tally types 11 and 12 write disk files containing coordinates of particles entering the zone. Type 11 writes 4 coordinates (X^2+Y^2, Z, velocity, and time), which can be used in (R, Z) symmetric systems. Type 12 writes 8 coordinates (X, Y, Z, alpha, beta, gamma, velocity, and time).

Tally type 16 writes the path length swept out in a zone, the time of entry, energy, zone number, batch number, and number of collisions. Tally type 17 writes the time of each collision and the energy lost in each collision.

Warning - any zone that you want tally type 11, 12, 16 or 17, binary output for **MUST** contain material. If it doesn't, the history will be terminated as soon as the particles enters the zone, before testing for any other type of tally.

Warning - the format of these binary files as used by TART differs from that used by the TARTNP code. For details of the contents of these files and FORTRAN routines that can be used to read them, contact the author.

Additional Output

In addition to the standard TART output the user may request additional output; almost anything can be scored and output.

edif j em(j) em(j+1) em(j+2).....

em(j), em(j+1), em(j+2).... are neutron tally type multipliers for tally energy groups **j, j+1, j+2,...**, respectively.

This option can be used to score and output anything that is proportional to the neutron tallies. For example, by inputting 50 cross sections (one for each of the standard 50 energy tally groups) for an activation reaction the results will be the activation due to this reaction. Similarly by inputting dose or damage data the results will be dose or damage. By inputting a simple detector response (simple in that it can only depend directly on the neutron energy), the output will be the detector response.

Restrictions: Using **edif** input automatically sets **sentl 28** to **2**, to indicate that the output will be multiplied by the **edif** multipliers for each neutron energy tally bin.

Example Input: The following input defines multipliers for the standard 50 neutron energy tally bins,

```
edif    1  4.0e-11 2.2e-11 8.0e-12 4.2e-12 2.3e-12 1.9e-12
edif    7  1.6e-12 1.3e-12 1.2e-12 1.1e-12 1.1e-12 1.1e-12
edif   13  1.2e-12 1.3e-12 1.7e-12 2.7e-12 4.4e-12 7.0e-12
edif   19  1.0e-11 1.4e-11 2.2e-11 3.1e-11 4.6e-11 6.4e-11
edif   25  8.0e-11 1.2e-10 1.7e-10 2.3e-10 3.2e-10 4.4e-10
edif   31  5.4e-10 6.0e-10 7.0e-10 9.2e-10 1.1e-09 1.3e-09
edif   37  1.5e-09 1.6e-09 1.8e-09 2.0e-09 2.3e-09 2.7e-09
edif   43  3.0e-09 3.3e-09 4.0e-09 4.1e-09 4.4e-09 5.0e-09
edif   49  6.0e-09 6.9e-09
```

edifg j em(j) em(j+1) em(j+2).....

em(j), em(j+1), em(j+2).... are photon tally type multipliers for tally energy groups **j, j+1, j+2,...**, respectively.

This option can be used to score and output anything that is proportional to the photon tallies. See **edif** above for a list of examples.

Restrictions: Using **edifg** input automatically sets **sentl 29** to **2**, to indicate that the output will be multiplied by the **edifg** multipliers for each photon energy tally bin.

Example Input: See **edif** above.

probm p1 p2 p3....

p1, p2, p3...are problem multipliers for energy deposited and fissions by zone and time step, in **MeV/gram**, for all zones. For each multiplier a separate output table will be produced.

Restrictions: Only one **probm** or **probmv** input line may be used in a problem, with up to 11 multipliers. **units** or **zonemult** (these two keywords are equivalent) may be used with **probm** or **probmv** to define multipliers by zone. For additional tally options using **probm** or **probmv**, see **sentl 35** and **36**, which allow additional scaling of output results by zone volume, energy tally group widths, etc..

Example Input: With the following input there will be three output tables, one without a multiplier, followed by a table using each of the two input multipliers.

```
probm   8.52e+4  48.56
```

probmv p1 p2 p3....

p1, p2, p3...are problem multipliers for energy deposited and fissions by zone and time step, in **MeV/cc**, for all zones. For each multiplier a separate output table will be produced.

Restrictions: Only one **probm** or **probmv** input line may be used in a problem, with up to 11 multipliers. **units** or **zonemult** (these two keywords are equivalent) may be used with **probm** or **probmv** to define multipliers by zone. For additional tally options using **probm** or **probmv**, see **sentl 35** and **36**, which allow additional scaling of output results by zone volume, energy tally group widths, etc..

Example Input: With the following input there will be three output tables, one without a multiplier, followed by a table using each of the two input multipliers.

probm 8.52e+4 48.56

reacted nz iza1 iza2.....

Requests additional edited output for zone **nz**, isotopes **iza1, iza2,...**(all isotopes if none are specified).

Restrictions: The tally type for zone **nz** MUST be **ltype 1** (expected collisions), **2** (path length) or **20** (Monte Carlo collisions); if not, its **ltype** will be set to **1**. If a **reacted** input line is used, the number of neutron tally groups is changed from 80 to 616. To suppress printing the entire array by energy, and to only print totals, set **sentl 23 1**.

Example Input: For zone 4 tally results of reactions with Li^6 (ZA = 3006),

reacted 4 3006

reactall nz1 thru nz2

Requests additional edited output for all isotopes in all zones from **nz1** through **nz2**, for which **ltype** has been set to **1, 2** or **20** by the use of **ltype** input.

Restrictions: **ltype** input MUST be used to define the tally type by zone; it is NOT sufficient to use **sentl 5** to set the tally type to **1, 2** or **20**. To suppress printing the entire array by energy, and to only print totals, set **sentl 23 1**.

Example Input: For reaction edits for all isotope in zones 3 through 30, use the following input,

reactall 3 thru 30

units zm nz1 nz2 nz3.....(0.0)

zonemult zm nz1 nz2 nz3.....(0.0)
units zm nz1 thru nz2 (0.0)
zonemult zm nz1 thru nz2 (0.0)

zm is a multiplier for the indicated zones. This multiplier is used with **probm** and **probmv** to give additional output for the zone. To obtain output involving products of **units/zoneunit** and **probm** or **probmv sentl 35** or **36** MUST be used to uniquely define the type of output requested.

Restrictions: The default is 0.0 = no additional output.

Changing Tally Bins

TART has standard options for tallying results as a function of energy and angle. The following options may be used to change these options, as well as to define time tally bins

angg - Change photon energy tally groups
angn - Change neutron energy tally groups
centim - Define time tally bins
cnang - Change neutron cosine tally bins
cpang - Change photon cosine tally bins
cneutal - Change neutron energy tally groups
cphotal - Change photon energy tally groups

angg j i(j) i(j+1) i(j+2).....

The default photon tally groups for tally types 7 through 10 are re-defined. The new energy tally group **j** has lower index **i(j)** and upper index **i(j+1)**, where **i(j)** and **i(j+1)** refer to the standard 51 tally group energy boundaries.

Example Input: The following input replaces the standard 50 tally groups. The new tally group 1, is from standard tally group boundary 1 through 3, new tally group 2, is from standard tally group boundary 3 through 6, new 3 is from 6 through 9, etc.

angg 1 1 3 6 9 12 14 17 18 20 22 24 26 28
angg 14 29 30 31 34 36 37 38 40 42 45 47 51

Restrictions: No more than 25 tally group indices can be specified. Tally group energy boundaries MUST be in ascending energy order.

angn j i(j) i(j+1) i(j+2).....

The default neutron tally groups for tally types 7 through 10 are re-defined. The new energy tally group **j** has lower index **i(j)** and upper index **i(j+1)**, where **i(j)** and **i(j+1)** refer to the standard 51 tally group energy boundaries.

Example Input: The following input replaces the standard 50 tally groups. The new tally group 1, is from standard tally group boundary 1 through 3, new tally group 2, is from standard tally group boundary 3 through 6, new 3 is from 6 through 9, etc.

```
angg   1  1  3  6  9  12  14  17  18  20  22  24  26  28
angg   14  29  30  31  34  36  37  38  40  42  45  47  51
```

Restrictions: No more than 25 tally group indices can be specified. Tally group energy boundaries MUST be in ascending energy order.

centim t(1) t(2) t(3).....[&]

t(1), t(2), t(3)....are the times (in shakes) at which various output quantities in their respective time intervals will be accumulated and edited. If needed, end the line with the continuation symbol (&) and use additional lines without **centim** to define additional times. The default value is **t(1) = 1.0e+8** (accumulate results integrated over all time).

Example Input: To accumulate and output results between 0 and 5 microseconds (0 and 500 shakes) and 5 and 10 (500 and 1000 shakes), and 10 and 1000 microseconds (1000 and 100,000 shakes),

```
centim   500   1000   1.0e+5
```

WARNING - Once particles reach the maximum census time they are terminated. Therefore this is an effective means of limiting how long particles are followed. However, users MUST insure that the initial time of all source particles does not exceed the maximum census time of the problem. Otherwise, results may be inconsistent.

cnang j c(j) c(j+1) c(j+2).....

Change the default values of the cosine bins for neutron tally types 7 through 10. The new bin **j** has limits **c(j)** and **c(j+1)**. The input values of **c** will be rounded to the nearest 0.01. No more than 20 **c** values (19 cosine bins) may be specified. The defined **c** MUST span the entire cosine range -1.0 to +1.0.

Example Input: To use smaller bins for the forward (cosine near +1) and backwards (cosine near -1) range use the following input,

```
cnang   1   1.0  0.98  0.96 0.94 0.92 0.9 0.88 0.8 0.7 0.6 0.5 0 -0.03
cnang  14   -0.5  -0.8  -0.9  -0.94  -0.96  -0.98  -1.0
```

cpang j c(j) c(j+1) c(j+2).....

Change the default values of the cosine bins for photon tally types 7 through 10. The new bin **j** has limits **c(j)** and **c(j+1)**. The input values of **c** will be rounded to the nearest 0.01.

No more than 20 **c** values (19 cosine bins) may be specified. The defined **c** MUST span the entire cosine range -1.0 to +1.0.

Example Input: see **cnang** below.

cneutal j i(j) i(j+1) i(j+2)....

Defines neutron tally groups that differ from the standard tally groups. The new energy tally group **j** has lower index **i(j)** and upper index **i(j+1)**, where **i(j)** and **i(j+1)** refer to the standard 51 tally group energy boundaries. The final energy on the input line defines the upper limit of the last group of the line. **i(1)** MUST be 1. See **sentl 46, reactall, reacted, ltype**.

Example Input: The follow maps the 617 neutron multigroup energy bounds (616 groups) into a new tally group structure,

```
cneutal   1   1 3 5 12 16 19 21 23 26 32 38 41 46 55 67 74 81 84
cneutal  19 89 91 94 96 99 101 102 103 104 105 107 109 111
cneutal   32 112 113 114 117 119 121 123 124 125 126 128 132
cneutal   44 133 137 141 143 150 157 167 217 317 417 517 617
```

cphotal j i(j) i(j+1) i(j+2)....

Defines photon tally groups that differ from the standard 50 tally groups. The new energy tally group **j** has lower index **i(j)** and upper index **i(j+1)**, where **i(j)** and **i(j+1)** refer to the standard 51 tally group energy boundaries. The final energy on the input line defines the upper limit of the last group of the line. **i(1)** MUST be 1. See **ltypeg**.

Example Input: The follow maps the 701 photon energy bounds into a new tally group structure,

```
cphotal   1   1 3 5 12 16 19 21 23 26 32 38 41 46 55 67 74 81 84
cphotal  19 89 91 94 96 99 101 102 103 104 105 107 109 111
cphotal   32 112 113 114 117 119 121 123 124 125 126 128 132
cphotal   44 133 137 141 143 150 157 166 201  301  401  501 701
```

Monte Carlo Test Calculations

TART includes a number of options to check the geometry and statistically estimate the volume of zones, including,

mcflood - Check for unassigned or overlapping spatial regions
mcvdisk - Statistically estimate the volume of zones
mcvplane - "

Checking for unassigned or overlapping spatial regions can now be done much more efficiently and effectively using the **TARTCHEK** interactive graphics code.

For spatial zones that are symmetric about the z axis TART can analytically calculate the volume of zones. Therefore **mcvdisk** and **mcvplane** need only be used for more general zones.

mcflood frac kgeom p1 p2 p3 p4 p5 p6 p7
mcflood frac 1 ri ro [z0 x0 y0 a1 a2]
mcflood frac 2 ri r0 z0 zlen [x0 y0]
mcflood frac 3 x1 x2 y1 y2 z1 z2

Check for unassigned or overlapping spatial regions in the volume defined by **kgeom** at an average sample density of **frac** per cc. Each point will be checked again all zones in a problem. If unassigned or overlapping space is found a message is printed. If 100 such cases are found the run will terminate.

If **kgeom = 1**, the volume is a segment of a spherical shell with inner radius **ri**, outer radius **ro**, centered at (**x0, y0, z0**), between angles **a1** and **a2** (degrees) from the z axis. **x0, y0, z0** are optional input that default to 0, 0, 0. **a1** and **a2** are optional input that default to 180 and 0 degrees.

If **kgeom = 2**, the volume is a right circular cylindrical shell parallel to the z axis, with inner radius **ri**, outer radius **ro**, with one end of cylinder centered at (**x0, y0, z0**), and length **zlen** along the positive z axis. **x0, y0** are optional input that default to 0, 0.

If **kgeom = 3**, the volume is a rectangular parallelepiped aligned with the three major axes and bounded by planes at **x1, x2, y1, y2, z1, z2**.

Recommendation: This option is no longer recommended. To check geometry use the interactive graphics program **TARTCHEK** which includes many more checks than those considered by **mcflood**.

Example Input: For a sphere of radius 970.1 cm, centered at the origin, check the volume where z < 0 (180 to 90 degrees from the z axis), using a sampling density of 0.0001 samples per cc.

mcflood 0.0001 1 0.0 970.1 0 0 0 180.0 90.0

mcvdisk frac ri ro w0 smax u0 v0 kaxis mzcheck nz
mcvdisk frac ri ro z0 smax x0 y0 0 mzcheck nz
mcvdisk frac ri ro x0 smax y0 z0 1 mzcheck nz
mcvdisk frac ri ro y0 smax z0 x0 2 mzcheck nz

An alternate spelling for this keyword is **mcvdisc**. The volume of zones is estimated statistically by launching particles in a positive direction parallel to the **w** axis (**x, y** or **z**),

from an annulus of a circular disk with inner radius **ri** and outer radius **ro**, centered at (**x0, y0 z0**), and perpendicular to the **w** axis at **w0**, to find the distance traversed in each zone. The particles will be randomly sampled at an average density **frac** over the surface of the annulus, and will traverse a maximum distance of **smax**, or until they reach a leakage zone. If **mzcheck = 1**, the zone and space check option is turned on, and tests are made for overlapping zones at the launch point and each zone boundary crossing. The search over zones will begin with zone **nz**, which may be 0 (default). Up to a combination of 27 **mcvdisk** and **mcvplane** input lines may be used. **Warning** - A combination of a large **frac** and **mzcheck** can be very time consuming.

Example Input: Use a disk perpendicular to the axis z (**kaxis = 0**) located at **z1 = -7.0**. The disk has an inner radius **ri = 0.0** and an outer radius **ro = 6.4** and is centered at **x0 = 0, y0 =0**. Traverse a maximum distance **smax = 8**. Check zones and space (**mzcheck = 1**). Sample **frac = 100** samples per unit area of the disk,

mcvdisk 100.0 0.0 6.4 -7.0 8.0 0.0 0.0 0 1

mcvplane frac u1 u2 v1 v2 w1 smax kaxis mzcheck nz
mcvplane frac x1 x2 y1 y2 z1 smax 0 mzcheck nz
mcvplane frac z1 z2 y1 y2 x1 smax 1 mzcheck nz
mcvplane frac x1 x2 z1 z2 y1 smax 2 mzcheck nz

The volume of zones is estimated statistically by launching particles in a positive direction parallel to the **w** axis (**x, y** or **z**), from a rectangular area perpendicular to the **w** axis at **w1**, and bounded by the limits **u1** to **u2** and **v1** to **v2** , to find the distance traversed in each zone. The particles will be randomly sampled at an average density **frac** over the surface of the rectangle, and will traverse a maximum distance of **smax**, or until they reach a leakage zone. If **mzcheck = 1**, the zone and space check option is turned on, and tests are made for overlapping zones at the launch point and each zone boundary crossing. The search over zones will begin with zone **nz**, which may be 0 (default). Up to a combination of 27 **mcvdisk** and **mcvplane** input lines may be used. **Warning** - A combination of a large **frac** and **mzcheck** can be very time consuming.

Example Input: Use a plane perpendicular to the axis z (**kaxis = 0**) located at **z1 = -7.0**. The plane extend between **x1 = -6.4** and **x2 = 6.4**, and **y1 = -6.4** and **y2 = 6.4**. Traverse a maximum distance **smax = 8**. Check zones and space (**mzcheck = 1**). Sample **frac = 100** samples per unit area of the rectangle,

mcvplane 100.0 -6.4 6.4 -6.4 6.4 -7.0 8.0 0 1

Collector Surfaces

Collector surfaces can be used to reposition and redirect particles. For example, consider a point isotropic source at the origin of our coordinate system and a small detector located a long distance away. If we track neutrons they will spread our over a spherical volume and the probability of a neutron actually reaching the detector can be very small.

By using a collector surface we can allow the neutrons to transport to a spherical surface with a radius slightly less than the radial distance to the detector. When the neutrons cross this spherical surface, if the zone that they enter defines a collector surface, we can reposition and redirect the neutrons toward the detector, thereby greatly increasing the probability of their actually reaching the detector.

Warning - Use collector surfaces with extreme care. Note, all particles entering a collector surface zone across any boundary are collected, and all particles are launched from only one side of the collector surface. Unless you carefully design your collector surface zones you can produce misleading results, that do not at all correspond to the situation you are trying to simulate.

It is imperative that collector plates ONLY be used in symmetric geometry, e.g., a simple spherical system, to avoid highly biasing results.

ltype (for neutrons) and **ltypeg** (for photons) tally 18 and 19 are used a collect particles entering a zone. You may collect particles in any number of zones in a problem. **t1819, t18sph** and **t18dpl** are used to define where to re-emit all collected particles. You may only use one set of **t1819, t18sph** and **t18dpl** in a problem.

t1819 nz r z0 n l1

If only **t1819** input used (no **t18sph** or **t18dpl** input) the collector surface is a circular disk perpendicular to the z axis, centered at the point (0, 0, **z0**), with radius **r**. All particles collected using **ltype** or **ltypeg** tally 18 or 19 are re-emitted in zone **nz**. For tally type 18, they are then given a random direction, but with the same angle from the surface norm as before, except the z component of the direction is positive. For tally type 19, they are directed in the positive z direction (no x or y component). The collector surface area is pi***r**^{2}.

n is a particle multiplier, i.e., for each particle collected **n** particles will be re-emitted. This will allow you to multiple otherwise small samples of particles. All zone numbers of **l1** or greater will have their output re-normalized by 1/**n** to return their results to the original source strength.

For a spherical segment collector surface use **t1819** and **t18sph** input; only allowed for tally type 18; see **t18sph**, below.

For a displaced spherical segment collector surface use **t1819** and **t18dpl** input; only allowed for tally type 18; see **t18dpl**, below.

Warning - Use this option with extreme care. Note, all particles entering any collector zone across any boundary are collected, and all particles are launched in zone **nz** from only one side of the collector surface.

Restrictions: the geometry must be spherically symmetric. Otherwise the rotation of particles to the collector is not valid.

t18sph [a1 a2 x0 y0]

The collector surface is a segment of a spherical surface. The sphere has a radius of **r**, and is centered by (**x0, y0, z0**). The surface segment is centered at (**x0, y0, z0 + r**), where **r** and **z0** are defined by **t1819** input. The surface segment extends between the angles **a1** and **a2** (degrees) from the positive z axis, and measured from the center of the sphere. The collector surface area is 2*pi*[Cos(**a2**) - Cos(**a1**)]***r**^{2}. All collected particles are re-emitted in zone **nz** (**nz** is defined by **t1819** input) at a random position on the collector surface. They are then given a random direction, but with the same angle from the surface as before, except that the component of the new direction in the radial direction of the spherical collector surface is positive.

Restrictions: **a1** and **a2** are optional input that default to 180 and 0 degrees, respectively. **x0** and **y0** are optional input that default to 0, 0. Note, the difference between **t18sph** which uses the radius **r**, on **t1819** input, and **t18dpl** which defines the radius based on (**x0, y0, z0**) and (**x1, y1, z1**). This option can only be used for tally type 18, and only for neutrons. **t1819** input is also required.

Warning - Use this option with extreme care. Note, all particles entering any collector zone across any boundary are collected, and all particles are launched in zone **nz** from only one side of the collector surface.

t18dpl a1 a2 x0 y0 x1 y1 z1

The collector surface is a segment of a spherical surface. The sphere has a radius of **r = Sqrt[(x1-x0)^2+(y1-y0)^2+(z1-z0)^2]**, and is centered by (**x1, y1, z1**). The surface segment is centered at (**x0, y0, z0**), where **z0** are defined by **t1819** input. The surface segment extends between the angles **a1** and **a2** (degrees) from the symmetry axis, which is from the center of the sphere to the center of the collector. The collector surface area is 2*pi*[Cos(**a2**) - Cos(**a1**)]***r**^{2}. All collected particles are re-emitted in zone **nz** (**nz** is defined by **t1819** input) at a random position on the collector surface. They are then given a random direction, but with the same angle from the surface as before, except that the component of the new direction in the radial direction of the spherical collector surface is positive.

Restrictions: **a1** and **a2** are optional input that default to 180 and 0 degrees, respectively. **x0** and **y0** are optional input that default to 0, 0. Note, the difference between **t18sph** which uses the radius **r**, on **t1819** input, and **t18dpl** which defines the radius based on (**x0, y0, z0**) and (**x1, y1, z1**). This option can only be used for tally type 18, and only for neutrons. **t1819** input is also required.

Warning - Use this option with extreme care. Note, all particles entering any collector zone across any boundary are collected, and all particles are launched in zone **nz** from only one side of the collector surface.

TARTNP Keywords Not Currently used by TART

The current version of TART does not have a restart capability **and currently there are no plans to add one**. This means that the following TARTNP keywords cannot currently be used by TART,

restart - Read a restart file to continue a calculation
fulltal - Read a restart file to perform a full edit

These options are described below, to document their former use, so that this information will not be lost from this newer TART documentation.

It is recommended that if you want to run more histories to improve statistics (the usual use of **restart**) you use the last random number seed printed at the end of your previous run, as input with **sentl 12** to run additional histories starting from exactly where the previous run ended. The results in the two (or more) output listing can then be added together to define your cumulative results.

restart file n

A previously run problem will be restarted from the restart file. If **n** is specified (any value), this input is treated a **fulltal** input (see, below). The complete problem input may consist of only **name, box, sentl 2, 11, 21** and **23** (only), **restart** (next to last line) and **end**. If **file** is omitted, the code will form the restart file name from the first 6 characters of the problem name on the **name** input line, which MUST be the same as for the original problem. When restarting a problem that was using tally type 11 or 12, the coordinate file that the code was filling when the previous run ended MUST be present. To **restart**, make a full edit, and then quit, see **fulltal**. Note, a restart file can only be used with exactly the same version of the code that created the restart file.

fulltal file

A previously run problem will be restarted from the restart file, a full edit will be made, and the run will then end. The complete input may consist of only **name, box, fulltal** and **end**. If **file** is omitted, the code will form the restart file name from the first 6 characters of the problem name on the **name** input line, which MUST be the same as for the original problem. Note, a restart file can only be used with exactly the same version of the code that created the restart file.

TARTNP Keywords Not Allowed by TART

Every effort has been made to make TART input parameters and output listing look exactly like the standard production code TARTNP input and results. However, a few TARTNP keywords have been judged to be very difficult to properly use and dangerous in the sense that they use can lead to erroneous and misleading results. Therefore the following TARTNP keywords are not allowed for use with TART,

chegp - Use fewer than the standard neutron groups
xsec - Change neutron multigroup cross sections
xsecg - Change photon interaction cross sections
xsecount - Change neutron multigroup cross sections

These options are described below, merely to document their former use, so that this information will not be lost from this newer TART documentation.

chegp j i(j) i(j+1) i(j+2).....

Use fewer than the standard neutron energy groups. The indices of the new groups **j, j+1,....** will be the original indices **i(j), i(j+1),....** The final **i** is the upper limit of the last group on the input line. The new set MUST include the original indices 1, 7, 25, 92, 114 and 176 (176 as the final **i**). If this option is used **cneutal** input MUST also be used to define neutron tally limits from the new (collapsed) neutron group structure.

This option to collapse groups has proven to be very dangerous to use and even the earlier TARTNP documentation recommended that it not be used.

xsec iso ireact j s(j) s(j+1) s(j+2).....[&]

The group average neutron cross section for isotope **iso** (in the form ZZAAA) and reaction **ireact**, starting with energy group **j**, will be replaced by the values **s(j), s(j+1),....** If needed, end the input line with the continuation indicator (&), and continuation another input line (with **xsec iso ireact j**) given more **s** values. See the table "Isotope Reaction Numbers" for a list of possible reactions. Note, this option does not apply when the Multiband option (**sentl 20**) is used.

This option to replace neutron multigroup cross sections has proven to be very dangerous to use. It is still included in TARTNP only for historical purposes, e.g., for compatibility with a few very old TARTNP input decks. These days if you want to change the cross sections it is easier to do while creating the TARTNP neutron interaction data file.

xsecg mat iso ir1 sm1 ir2 sm2 ir3 sm3

The photon interaction cross sections for material **mat**, isotope **iso** (in the form ZZAAA), and internal reaction numbers **ir1, ir2** and **ir3**, will be multiplied by **sm1, sm2** and **sm3**, respectively. The internal reaction number is: 1 for incoherent, 2 for coherent, 3 for pair production and 4 for photoelectric. No more than 19 **xsecg** input lines may be used.

This option to replace photon interaction cross sections has proven to be very dangerous to use. It is still included in TARTNP only for historical purposes, e.g., for compatibility with a few very old TARTNP input decks. These days if you want to change the cross sections it is easier to do while creating the GAMDAT photon interaction data file.

xsecount iso icount j s(j) s(j+1) s(j+2).....[&]

The group average neutron cross section for isotope **iso** (in the form ZZAAA) and reaction count **icount**, starting with energy group **j**, will be replaced by the values **s(j), s(j+1),....** If needed, end the input line with the continuation indicator (&), and continuation another input line (with **xsecount iso icount j**) given more **s** values. See the table "Isotope Reaction Numbers" for a list of possible reactions. Note, this option does not apply when the Multiband option (**sentl 20**) is used.

This option to replace neutron multigroup cross sections has proven to be very dangerous to use. It is still included in TARTNP only for historical purposes, e.g., for compatibility with a few very old TARTNP input decks. These days if you want to change the cross sections it is easier to do while creating the TARTND neutron interaction data file